

# Exploring 3D Shapes through Real Functions

Submitted by

Silvia Biasotti

*Università degli studi di Genova*  
*Dipartimento di Informatica,*  
*Sistemistica e Telematica*



*CNR - Istituto di Matematica Applicata e*  
*Tecnologie Informatiche – Sez. di Genova*



**Dottorato di Ricerca in Scienze e Tecnologie  
dell'Informazione e della Comunicazione  
(XX ciclo)**

**Indirizzo: Ingegneria Elettronica e Informatica**  
**SSD: ING-INF 05**

*Dissertation*

**Exploring 3D Shapes through Real Functions**

*Submitted by*  
***Silvia Biasotti***

*Supervisors*

**Dr. *Bianca Falcidieno***  
*Istituto di Matematica Applicata e Tecnologie Informatiche*

**Dr. *Michela Spagnuolo***  
*Istituto di Matematica Applicata e Tecnologie Informatiche*

April, 2008

# Sommario

Questa tesi si colloca nell'ambito di ricerca riguardante la rappresentazione, la modellazione e la codifica della conoscenza connessa a *forme digitali*, dove per *forma* si intende l'aspetto visuale di ogni oggetto che esiste in due, tre o più dimensioni. Le forme digitali sono rappresentazioni di oggetti sia reali che virtuali, che possono essere manipolate da un calcolatore.

Lo sviluppo tecnologico degli ultimi anni in materia di hardware e software ha messo a disposizione una grande quantità di strumenti per acquisire, rappresentare e processare la geometria degli oggetti; tuttavia per gestire questa grande mole di dati è necessario sviluppare metodi in grado di fornirne una codifica efficiente.

In questa tesi si propone un modello concettuale che descrive un oggetto 3D attraverso la codifica delle caratteristiche salienti e ne definisce una bozza ad alto livello, tralasciando dettagli irrilevanti. Alla base di questo approccio è l'utilizzo di descrittori basati su funzioni reali in quanto forniscono un'astrazione della forma molto utile per analizzare e strutturare l'informazione contenuta nel modello discreto della forma. Una peculiarità di tali descrittori di forma è la capacità di combinare proprietà topologiche e geometriche consentendo di astrarne le principali caratteristiche. Per sviluppare questo modello concettuale, è stato necessario considerare gli aspetti sia teorici che computazionali relativi alla definizione e all'estensione in ambito discreto di vari descrittori di forma.

Particolare attenzione è stata rivolta all'applicazione dei descrittori studiati in ambito computazionale; a questo scopo sono stati considerati numerosi contesti applicativi, che variano dal riconoscimento alla classificazione di forme, all'individuazione della posizione più significativa di un oggetto.



# Abstract

This thesis lays in the context of research on representation, modelling and coding knowledge related to digital shapes, where by *shape* it is meant any individual object having a visual appearance which exists in some two-, three- or higher dimensional) space. *Digital shapes* are digital representations of either physically existing or virtual objects that can be processed by computer applications.

While the technological advances in terms of hardware and software have made available plenty of tools for using and interacting with the geometry of shapes, to manipulate and retrieve huge amount of data it is necessary to define methods able to effectively code them.

In this thesis a conceptual model is proposed which represents a given 3D object through the coding of its salient features and defines an abstraction of the object, discarding irrelevant details. The approach is based on the shape descriptors defined with respect to real functions, which provide a very useful shape abstraction method for the analysis and structuring of the information contained in the discrete shape model. A distinctive feature of these shape descriptors is their capability of combining topological and geometrical information properties of the shape, giving an abstraction of the main shape features. To fully develop this conceptual model, both theoretical and computational aspects have been considered, related to the definition and the extension of the different shape descriptors to the computational domain.

Main emphasis is devoted to the application of these shape descriptors in computational settings; to this aim we display a number of application domains that span from shape retrieval, to shape classification and to best view selection.



To Paolo





# Acknowledgements

There are lots of people I would like to thank for a huge variety of reasons.

I especially thank my supervisors: Bianca Falcidieno, for convincing me, in 2005, to start this course, and Michela Spagnuolo who encouraged me during these years. I have worked with them for the last ten years and they drove me in research with expert guidance.

This thesis also benefited from many insights, questions, and suggestions from Massimo Ferri and all people at the Computer Vision Group of the Dipartimento di Matematica of the Università di Bologna.

I would like to thank the researchers with whom I had the privilege and the pleasure to work, in particular Patrizio Frosini, Claudia Landi, Andrea Cerri for ARCES, Università di Bologna; Leila De Floriani, Laura Papaleo of the Dipartimento di Scienze dell'Informazione of the Università di Genova; to Gabriella Sanniti di Baja of the Istituto di Cibernetica E. Caianello, CNR in Pozzuoli, Napoli; to Remco Veltkamp from the Institute of Information and Computing Sciences, Utrecht University; to Dominique Attali from the Laboratoire des Images et des Signaux, CNRS in Grenoble, France; to Herbert Edelsbrunner from the Computer Science and Mathematics Dept. at Duke University, USA; to Craig Gotsman, Gershon Elber, Gill Barequet and Oleg Polonsky, from the Center for Graphics and Geometric Computing at TECHNION in Haifa, Israel and to all other coauthors. I appreciate their interest and encouragement more than they probably realize.

A special thank goes also to my colleagues and coauthors Marco Attene, Daniela Giorgi, Simone Marini, Michela Mortara and Giuseppe Patané from IMATI, CNR in Genova. Thanks to Laura Paraboschi, my coauthor and first graduate student.

Moreover, I thank all people of the Istituto di Matematica Applicata e Tecnologie Informatiche of the Consiglio Nazionale delle Ricerche for the friendly atmosphere in the department, in particular I list Riccardo Albertoni who shares the office with me, Chiara Catalano for the kind friendship and the encouragements, Daniele D'Agostino a “wizard” to decode binary files, Corrado Pizzi for his helpful technical support, Marinella Pescaglia and Sandra Burlando for their first class management of all administrative stuffs, Antonella Galizia, Elena Camossi, Maria Grazia Ierardi, Francesco Robbiano, and the other graduate students

with whom I lunch every day: I had a very enjoyable time!.

Last but not certainly least, I would like to thank my family and friends for their love and patience. A special thank goes to Paolo who suffered me during these years.

This work has also been partially supported by the European FP6 Network of Excellence “*AIM@SHAPE*”, contract number 506766, by the Italian National Project “*SHALOM*” funded by the Italian Ministry of Research under contract number RBIN04HWR8, by the CNR research grants *Topologia e omologia nell’analisi di forme geometriche 3D*, grant DG.RSTL.050.004 and *Tecniche avanzate per l’analisi e la sintesi di forme digitali 3D*, grant ICT.P10.

# Table of Contents

<b>List of Figures</b>	ix
<b>List of Tables</b>	xvii
<b>Introduction</b>	<b>1</b>
Motivation	3
Contributions	4
Overview	6
<b>Chapter 1 Background notions</b>	<b>9</b>
1.1 Mathematical background	9
1.2 Computational background	12
1.3 Real functions	14
<b>Chapter 2 Describing shapes using real functions</b>	<b>17</b>
2.1 Morphology-based shape descriptors	20
2.1.1 Morse complex	21
2.1.2 Morse-Smale complex	25
2.1.3 Computational complexity	30
2.1.4 Applications	31
2.2 Graph-based shape descriptors	33
2.2.1 Contour tree	34

2.2.2	Reeb graph . . . . .	42
2.2.3	Computational complexity . . . . .	50
2.2.4	Applications . . . . .	52
2.3	Algebraic shape descriptors . . . . .	55
2.3.1	Size theory . . . . .	55
2.3.2	Persistent homology . . . . .	62
2.3.3	Morse descriptor . . . . .	67
2.3.4	Computational complexity . . . . .	68
2.3.5	Applications . . . . .	69
2.4	Shape descriptors based on distance transform . . . . .	73
2.4.1	Medial axis . . . . .	73
2.4.2	Shock graph . . . . .	75
2.4.3	Centrelines and Curve skeleton . . . . .	77
2.4.4	Computational complexity . . . . .	79
2.4.5	Applications . . . . .	79
2.5	Discussions and comparative remarks . . . . .	82
2.5.1	Overall comparison and general remarks . . . . .	83
2.5.2	Expressiveness of shape descriptors . . . . .	85
2.5.3	Suitability for applications . . . . .	88
<b>Chapter 3</b>	<b>Contribution . . . . .</b>	<b>91</b>
3.1	The Shape Graph . . . . .	92
3.1.1	Coupling the graph representation with geometric attributes . . . . .	93
3.1.2	Extension to sets of objects . . . . .	99
3.2	Higher-dimensional size functions . . . . .	101
3.2.1	Size functions of 3D objects from topological graphs . . . . .	102
3.2.2	Multidimensional size functions . . . . .	107
3.3	Comparison of real functions . . . . .	109

3.3.1	Continuous case . . . . .	111
3.3.2	Discrete case . . . . .	112
3.4	Discussions . . . . .	116
<b>Chapter 4</b>	<b>Applications and Results . . . . .</b>	<b>119</b>
4.1	Shape Matching and Retrieval . . . . .	120
4.1.1	Shape retrieval using high dimensional size functions . . . . .	121
4.1.2	Sub-part correspondence . . . . .	134
4.1.3	Scene comparison . . . . .	142
4.2	Shape Classification . . . . .	148
4.2.1	Creative prototypes using structural descriptors . . . . .	152
4.2.2	Evaluation . . . . .	154
4.3	Best view selection . . . . .	159
4.3.1	View descriptors . . . . .	160
4.3.2	Sampling the view space . . . . .	165
4.3.3	Experimental Results and discussions . . . . .	168
4.4	Discussions . . . . .	169
<b>Chapter 5</b>	<b>Conclusions . . . . .</b>	<b>171</b>
5.1	Summary of results . . . . .	171
5.2	Directions for future work . . . . .	174
<b>Bibliography</b>	<b>. . . . .</b>	<b>177</b>



# List of Figures

1	Even if geometrically non equivalent, all these chair models have a similar structure and the human brain groups them in the same object class. . . . .	1
2	The peaks, pits and passes of the model in (a) may be formal expressed in mathematical terms as critical points (maxima, minima and saddles) (b). . .	2
3	Shape matching framework: one or more shape descriptors, or signatures, are associated to a 3D model; then a distance between shape descriptors is defined.	5
1.1	Examples of 0-, 1-, 2- and 3-simplices. . . . .	10
1.2	Configuration of vertices around a maximum, saddle, and monkey saddle point.	14
1.3	(a) Height function, (b) Euclidean distance from the barycenter, (c) average geodesic distance in [HSKK01], (d) distance from curvature extrama, (e) first eigenfunction of the Laplacian matrix of the model. . . . .	16
2.1	(a) The ascending Morse complex of a two-dimensional scalar field (the 2-cells correspond to the minima). (b) The Morse-Smale complex. Its 1-skeleton (the set of simplices of dimension 0 and 1) is the critical net. . . . .	23
2.2	(a) The integral lines emanating from the higher saddle $s$ reach $s'$ , but a slight perturbation of the height direction causes the integral lines to reach $m$ instead (b). The Morse complex is shown in (c). In (c) the black vertex correspond to the 0-cell, the blue lines represent the 1-cells while the light blue region is the 2-cell. . . . .	26
2.3	The four possible configurations for the slope districts in a CPCG. . . . .	27
2.4	The normal space for a cluster of triangles incident on a vertex. . . . .	28

2.5	(a) Visualization of the bubble structures in a Rayleigh-Taylor instability problem [LBM <sup>+</sup> 06]. (b) The initial Morse-Smale complex of spatial probability distribution of electrons in a hydrogen atom under a large magnetic field and the Morse-Smale complex features after simplification.[BPH05] . . . . .	31
2.6	Isosurfaces around a minimum (a) and a saddle (b-c). At the saddle point in (d), a torus evolves into a sphere changing the genus of the isosurface without altering the number of components (1) of the level set. . . . .	35
2.7	A two-dimensional scalar field (a,b) and its contour tree (c,d). The edge orientation and the spatial embedding of the contour tree are shown in (d). .	36
2.8	The join tree and the split tree (left) of the contour tree (right) of the scalar field in Figure 2.7. . . . .	38
2.9	Reeb graph with respect to the height function (a) and to the distance from a point (b). . . . .	44
2.10	Pipeline of the multi-resolution Reeb graph extraction in [HSKK01]. . . . .	46
2.11	Level sets (a) and the centerline (b,c) of an horse using the geodesic distance from a source point as proposed in [Lazarus and Verroust 1999]. . . . .	49
2.12	(a) Vertex classification based on Gaussian curvature; (b) high curvature regions are depicted in red; (c) topological rings expanded from centers of high curvature regions; (d) the graph obtained as proposed in [Mortara and Patané 2002]. . . . .	50
2.13	Contour tree may be used to support interactive exploration of structures that are hidden in the conventional view [CSvdP04]. . . . .	52
2.14	Surface parameterization using the Extended Reeb Graph. (a,b) A topology based decomposition of the shape derived from the ERG is used to define a chart decomposition of the mesh, and each chart is parameterized with respect to the cuts shown in (b); (c) the normal-map images. . . . .	54
2.15	(a) The size pair $(S, f)$ , where $S$ is the curve represented by a continuous line and $f$ is the function “distance from the point $P$ ”. (b) The size function of $(S, f)$ . . . . .	57
2.16	Two size functions can be described by cornerpoints and cornerlines and compared by the matching distance. . . . .	58
2.17	The discretization of a space described in [Fro92]. (a) The original space. (b) The process of $\delta$ -covering. (c) The approximating graph. . . . .	60
2.18	The persistent homology of a filtered complex can be represented by $\mathcal{P}$ -intervals.	64



2.19	(a) Top view of the molecular surface for Gramicidin A, presenting a channel as primary topological feature. (b) Filtration of a complex representing the molecular data. (c) Graph of $\beta_1^{l,p}$ showing that eliminating 1-cycles with low between the feature of the representation. (Provided Zomorodian.) . . . . .	71
2.20	Medial axis of two planar shapes. In the second example the medial axis is shown also for the external part of the shape. . . . .	74
2.21	The medial axis (a) and the shock graph (b) of two simple curves. . . . .	76
2.22	(a) Image matching using shock graphs. (b) House path planning using approximated skeleton. Cruve skeleton used as central path for virtual bronchoscopy (c) and colonoscopy (d). . . . .	81
2.23	(a,b) Two surfaces studied with respect to the height function $f$ in the horizontal direction; the values of $f$ are depicted on the arrows. (c,d) Their corresponding Reeb graphs are not isomorphic, and therefore allow to distinguish the original shapes. (e,f) The 0th and the 1st persistent homology groups of the two surfaces coincide, and therefore are not sufficient to discriminate between these objects. . . . .	87
3.1	The distance from the barycenter is highlighted on the hand model (a). The SG nodes correspond to the regions generated from the contours (b). In (c), the SG is superimposed to the model, while the representation in (d) shows all the SG nodes. . . . .	92
3.2	Some segments associated to the size graph nodes (graph obtained by segmenting the mesh with the insertion of seven level sets). . . . .	94
3.3	Details about the geometric meaning of some attributes. (a) Minimum, maximum and average radius of a region. (b) $A_1$ and $A_2$ represent two pseudo-conic areas. (c) Lenght of upper and lower (with respect to a chosen $f$ ) boundary components. . . . .	95
3.4	SG nodes are associated to model regions (a). The SG in (b) is simplified and edges are oriented according to the increasing values of the distance from the barycenter (c). . . . .	96
3.5	(a) The SG representation of the model in Figure 3.4 and some of the model sub-parts associated to graph nodes. (b) The structural descriptor of a human body model . . . . .	97
3.6	Models having different size and resolution. . . . .	98

3.7	Two possible connections between a virtual node $X$ and a scene made by two graphs. Provided that, for each graph, $X$ is connected to a node of minimum degree, the representation of the scene graph is independent of the choice of $X$ (unless of graph isomorphisms). . . . .	100
3.8	(a) Two different representations of a cactus shape and (b) their corresponding size functions. . . . .	102
3.9	(a,b) Minimal bounding boxes of two models. . . . .	103
3.10	Changes due to a simplification of a model do not significantly alter the Shape Graph and its size function. (a) A model and its simplified version; (b) their size graphs with some attributes highlighted, using the integral geodesic distance $f^3$ and the measuring function <i>area</i> ; (c) the corresponding size functions, which show very small variations. . . . .	105
3.11	Example of the computation of a two-dimensional size function on two surface models, on five half-planes of the domain partition defined in [BCF <sup>+</sup> 07]. . . .	109
3.12	(a) Iso-contours of two functions $f$ and $g$ that intersect at $p$ . (b) Discretization of the gradient field of $f$ at $p_i$ with respect to its 1-star. . . . .	111
3.13	Color image of the matrix $\mathbf{A}$ related to the first 50 eigenfunctions of $D$ ; on the left (resp., right) pairs of Laplacian eigenfunctions $(f_1, f_2)$ (resp., $(f_2, f_6)$ ) with the lowest (resp., highest) dissimilarity measure $\overline{\mathcal{I}}$ . . . . .	113
3.14	(a-b) Variation of $\overline{\mathcal{I}}$ on $D$ for the pairs of functions in Figure 3.13; moving from white to pink the dissimilarity of the functions increases. (c) Iso-contours of the function orthogonal to $f_1$ ; (d) critical points of $f_1$ and visualization of $\overline{\mathcal{I}}$ . . . .	114
3.15	The picture shows the co-domain and iso-contours of the function $g$ orthogonal to a given $f$ everywhere on $D$ with the exception of the critical points of $f$ . . . .	115
4.1	Precision-recall diagrams on the training dataset, involving different SG representations and attributes. (a) The mapping function to extract the centerline is the normalized integral geodesic distance [HSKK01]; the measuring function varies in a set of four attributes. (b) The mapping functions varies, while the measuring function is always the minimum radius of a region. . . . .	122
4.2	Comparison with existing retrieval methods. Recall histograms: the values are averaged on the whole database. . . . .	123
4.3	Comparison with existing retrieval methods. (a) average rank and (c) last place ranking. . . . .	124
4.4	Precision-recall diagrams for the different classes in our database. . . . .	125

4.5	Retrieval performance of our method over a database of 280 models, the MRG [HSKK01] and the Spherical Harmonics [KFR03]. . . . .	129
4.6	1-dimensional size functions of three models over the half-planes of $\mathbb{R}^2 \times \mathbb{R}^2$ defined by the parameters ( $\theta = \frac{\pi}{12}$ , $\theta = \frac{\pi}{4}$ , $\theta = \frac{11\pi}{12}$ ) and $\vec{b} = (0, 0)$ . . . . .	131
4.7	Top retrieval results when four single measuring functions and the 3-dimensional size function that combines $\varphi_x$ , $\varphi_y$ and $\varphi_z$ are used. Results are depicted in every column in increasing order of distance from the first model. . . . .	132
4.8	Query results according to different choices of the mapping and the measuring functions. (a) Emphasis on the spatial position, using $f^1$ and $\varphi_{P_5, P_6}$ ; (b) emphasis on the fatness, with $f^1$ and the <i>area</i> of the regions; (c) humans in different poses are recognized, using $f^3$ and $r_{av}$ ; (d) results on the class of four-limbs reflecting perceptual similarity, obtained by $f^4$ and $r_{max}$ . . . . .	133
4.9	The two models and their graphs (a). The node correspondence is represented by nodes with the same color (b), and by regions of the same color. . . . .	135
4.10	The sub-part correspondence between the models (a) and (b) is shown associating the same color to regions that are matched. Regions in grey are not matched. Similar results for (c) and (d). . . . .	136
4.11	The model (b) has two small holes on its rear part that do not have any counterpart in the model (a). A detail of the correspondence between sub-parts the two models is shown in (c): the grey region identifies the difference between the two models. . . . .	137
4.12	Correspondence between the sub-parts of two chairs (a), two glasses (b), two hands (c) and a human model in three different poses (d). . . . .	138
4.13	The correspondence between the sub-parts of the two models is highlighted by the color. . . . .	138
4.14	Two models are shown together with their graphs (a). The correspondence between the two graphs is represented by nodes with the same colors (b). The node correspondence between the two graphs is reproduced on the two models, respecting the colors associated to the nodes (c). . . . .	139
4.15	The models (b) and (d) are recognized as sub-parts of the models (a) and (c) respectively. . . . .	139
4.16	Some examples of partial correspondences. . . . .	140
4.17	(a) SHREC database, containing 400 models divided into 20 classes. Each row represents a class of objects. . . . .	141

4.18	The set of queries . Each query is obtained by composing sub-parts of objects belonging to the dataset. . . . .	142
4.19	Normalized Discount Cumulated Gain considering only highly relevant models (a) and both highly relevant and marginally relevant models (b). . . . .	143
4.20	Scenes with a man and a teddy bear: distance values when using height function	145
4.21	Scenes with one chair . . . . .	145
4.22	The values of the distance $D_N$ between these two scenes is generally high; this means that the two scenes are not globally similar. . . . .	146
4.23	(a) Query, (b) the database and the distances from $Q$ , and (c) the database ordered according to the distance . . . . .	147
4.24	Scenes with a man and a table . . . . .	147
4.25	A false positive result . . . . .	148
4.26	Precision-recall diagrams on the four classes of <i>Data1</i> . . . . .	149
4.27	Precision-recall diagrams on the four classes of <i>Data2</i> . . . . .	150
4.28	Overview of the classification process when class prototypes are considered. .	152
4.29	A set of models (upper row) and their SG descriptor (lower row) and the class prototype (right). . . . .	154
4.30	Details on the geometric attributes of the prototype obtained in figure 4.29. .	154
4.31	80 models used as queries when evaluating the classification framework. . . .	155
4.32	Comparison among the classification rates using one element (top), two and three elements (resp. middle and bottom) per class from selectively and randomly chosen representants and our Creative prototypes. The abscissa is the ratio between the size of the training set and the number of queries. . . . .	157
4.33	Improvement of a query answer when pre-classification is involved in the retrieval pipeline. . . . .	158
4.34	False positive retrieval results, averaged over 80 queries against 400 models. Value $0.y$ means $y\%$ percentage of false positives, and is plotted vs the number of classes taken into account after pre-classification. . . . .	159
4.35	Four top-ranking (left to right) views among candidate views according to the surface area entropy descriptor. Black dots indicate a three-quarter view (otherwise it is a normal clustered view). . . . .	161

4.36	Four top-ranking (left to right) views among candidate views according to the <i>visibility ratio</i> descriptor. . . . .	162
4.37	Four top-ranking (left to right) views among candidate views according to the <i>curvature entropy</i> descriptor. Black dots indicate a three-quarter view (otherwise it is a normal-clustered view). . . . .	163
4.38	Four top-ranking (left to right) views among candidate views according to the <i>silhouette length</i> descriptor. Black dots indicate a three-quarter view (otherwise it is a normal-clustered view). . . . .	164
4.39	Four top-ranking (left to right) views among candidate views according to the <i>silhouette entropy</i> descriptor. Black dots indicate a three-quarter view (otherwise it is a normal-clustered view). . . . .	165
4.40	Four top-ranking (left to right) views among candidate views according to the <i>topological complexity</i> descriptor. Black dots indicate a three-quarter view (otherwise it is a normal-clustered view). . . . .	166
4.41	Four top-ranking (left to right) views among candidate views according to the <i>surface entropy of semantic parts</i> descriptor. Black dots indicate a three-quarter view (otherwise it is a normal-clustered view). . . . .	167
5.1	The Shape Annotator is able to annotate a shape adopting different segmentations [ARSF07]. . . . .	173
5.2	The Shape Graph in (a) drives the shape segmentation with iso-contours and flow lines (b). (c) The region boundaries are sampled. (d) The rough triangle mesh that approximates the original model on the basis of the point sampling.	173



# List of Tables

2.1	Algorithms for the extraction of the descending/ascending Morse complex or of the Morse-Smale complex. For each algorithm the input model is outlined (regular or simplicial) as well as the output it produces (Morse or Morse-Smale complex). . . . .	30
2.2	Classification of the methods that extract graph-based descriptors. Symbols: $N$ is the number of higher-dimensional simplices or cells; $n$ is the number of vertices or points; $m$ is the number of vertices inserted in the mesh during contouring phases; $c$ is the number of critical points; $C$ is the number of tree nodes; $\alpha$ is the inverse of the Ackermann function; $k$ is the length of the longest path traversed in the tree, $p$ is the number of pixels, $v$ is the number of voxels $t$ is time, $e$ is the number of edges in the neighborhood tree and $E$ is the amount of birth-death and interchange events. Note that, in the 2D case, $N = O(n)$ . . . . .	51
2.3	Computational costs of descriptors based on algebraic theory. Symbols: $n$ represents the number of vertices or points or pixels (voxels); $m$ corresponds to the number of edges; $n'$ is the number of vertices of the size graph; $N$ is the number of simplices or complexes. . . . .	68
2.4	Classification of the methods for skeleton extraction. Symbols: $n$ represent the number of vertices or points or pixels (voxels); $m$ the number of vertices inserted in the mesh during an eventual contouring phase; $e$ the number of edges in the neighborhood tree, $r$ is the number of reflex vertices. . . . .	79
3.1	Statistics for the models in Figure 3.6. . . . .	98
3.2	Values for the matching distances between six different models in our database. The size graphs have been obtained using the integral geodesic distance $f^3$ and the region area. . . . .	106

3.3	Statistics for some models, relating the dimension of the original model with the number of nodes ( $\#N$ ) of the size graph $G^f$ , the number of cornerpoints ( $\#CP$ ) of the corresponding size function SF and the storage size of the final descriptor. . . . .	107
3.4	Time requirements for the computation of the size function of some 3D images of different dimensions. $ V $ and $ E $ represent the number of vertices and edges of the size graphs of the models. Avg. time is the average time required to compute the size function on a single half-plane of the foliation, while Total time refers to the computation of the size functions on 9 half-planes. Analogously, Avg. $ C $ is the average number of cornerpoints of the size function on a single half-plane of the foliation, and Total $ C $ is the sum of the number of cornerpoints of the size functions on 9 half-planes. . . . .	110
4.1	Distances between models: the 2-dimensional matching distance between the 2-dimensional size functions associated to $\vec{\varphi}_{(1,-1)}$ (top value) and the maximum between the 1-dimensional matching distances associated to the 1-dimensional measuring functions $\varphi_1$ and $\varphi_{-1}$ (bottom). . . . .	128
4.2	Multidimensional and 1-dimensional matching distance between an airplane and another airplane (first column) or a chair model (second column). The first three rows refer to the multidimensional matching distance using 3 different singular half-planes of the foliation, the fourth and fifth row yield the average and the maximum value of the multidimensional matching distance using 9 planes, and the last three rows represent the 1-dimensional matching distance using the single components of $\vec{f}$ . . . . .	130
4.3	Matching distances between six different models in our database over a single plane of a foliation. Computing $420 \times 420$ comparisons between the models in the database requires 9.68s. . . . .	131
4.4	Classification rate indicates the percentage of query models which are correctly classified. . . . .	155

---



# Introduction

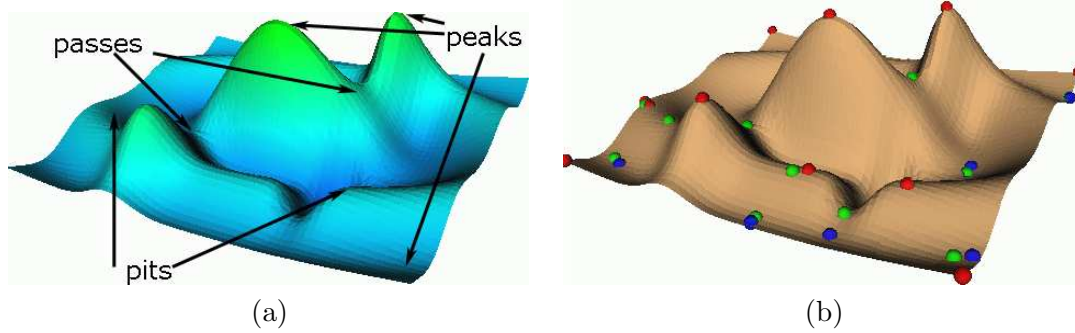
Shape recognition and classification are basic steps to capture and understand the shape of spaces, identifying how they match and differ in shape. In this context shape descriptions are the starting point to define representation frameworks available for graphics and design. To make this task computationally affordable, shapes are transformed in some fixed way to obtain a description which is both easily computable and able to keep the main shape properties.

An important issue of shape descriptions is their capability of decomposing the model into sub-parts, each presumably simpler to describe than the original one and, at last, combining these sub-descriptions in a global description. This approach is supported by cognitive research, for instance the human perception theories proposed by Marr [Mar82] and Biedermann [Bie87, Bie95], where experimental results are used to show that people, when interpreting the meaning of a novel scene, attend only to a few details and recognize an object on the basis of basic level shapes called *geons*. Finally, a representation that specifies parts (geons), attributes and relations between them, independently and explicitly is called a *structural decomposition*. In Figure 1, we show a list of shapes that the human intuition refers to the same basic level shape: a chair model; these models are not geometrically equivalent but they share a common structure, which can be summarized by a seat, a rear part, etc.



**Figure 1:** Even if geometrically non equivalent, all these chair models have a similar structure and the human brain groups them in the same object class.

To set the computational framework for shape description which is the purpose of this dissertation, it is necessary to establish which are the most relevant characteristics with respect



**Figure 2:** The peaks, pits and passes of the model in (a) may be formal expressed in mathematical terms as critical points (maxima, minima and saddles) (b).

to the application domain; to define the features in which the object will be decomposed, and to define the mathematical/computational framework that identifies them; to develop algorithms for the extraction of salient features; to encode the achieved decomposition and construct an explicit structure as *shape descriptor*.

Computational topology may give a theoretical framework for the formalization and solution of problems related to shape and shape understanding [VY90, Veg97, DEG99]. Since the beginning, the importance of topological aspects in modeling and analysis has been recognized in many computer application areas [Fro90, Har99, Axe99, BFS00, EHZ03, BFD<sup>+</sup>08]. For instance, there is a parallelism between the salient features of a terrain (e.g. peaks, pits and passes) and the critical points with respect to the height direction of the corresponding digital terrain model (see Figure 2). In general, computational approaches to topological questions deal with the complexity of issues related to the study of invariants under continuous deformations such as the number of connected components, holes, tunnels, or cavities and with the design of efficient algorithms for solving them.

This thesis deals with the analysis and the definition of an automatic and computationally affordable shape description framework available for shape understanding and coding. The main emphasis is devoted to its application in computational settings; to this aim we will display a number of application domains that span from shape classification, retrieval and matching to best view selection. In the remainder of the introduction, we firstly organize and motivate our work, then an overview of the main contribution of this thesis to the research in Shape Modeling is proposed; finally, the description of the organization of the following chapters ends this part.

## Motivation

The major role of Computer Graphics and Computer Vision is the study of basic models and methods to represent, generate and analyze shapes. At the beginning, Computer Graphics mostly focused on solving basic problems related to representation issues of subspaces of the Euclidean 3D-space  $\mathbb{R}^3$  [Req80, M88]. Then, the increasing popularity of videos and animations coming from real (e.g. medical) and virtual worlds (e.g. cartoons, animation of virtual characters) makes necessary the use of time-varying and higher dimensional data. In Computer Vision, less emphasis has been put on representation issues, as in this field digital representations are generally limited to the pixel-, or voxel-based encoding of objects acquired from the real world. On the other hand, researchers in Computer Vision introduced the fundamental idea of using compact representations of shapes, namely *shape descriptors*, and addressed issues related to analyzing, understanding and recognizing objects.

More recently, we have seen a gradual shift of research interests from methods to represent shapes towards methods to *describe* shapes in Computer Graphics as well. While a digital model, either pixel- or vector-based, is a digital representation that is quantitatively similar to an object, its description is only qualitatively similar. The distinction between *representation* and *description* can be expressed as follows [Nac84]:

an object representation contains enough information to reconstruct (an approximation to) the object, while a description only contains enough information to identify an object as a member of some class.

The representation of an object is thus more detailed and accurate than a description, but it does not necessarily contain any high-level information on the shape of the object explicitly. The description is more concise and conveys an elaborate and composite view of the object class. Therefore, the concept of description implicitly refers to an important aspect of modeling: the analysis and the simplification of the representation.

The reason for the shift of interest from representations to descriptors is the need of rapidly and effectively extracting knowledge from massive volumes of digital content and the demand of handling new forms of content, such as 3D animations and virtual or augmented reality. While the technological advances in terms of hardware and software have made available plenty of tools for using and interacting with the geometry of shapes, there is an increasing demand of methods for the automatic extraction of the structure and the semantic content of digital shapes and the generation of shape models to be retrieved, shared, exploited and used to construct new knowledge [IST]. While research in modeling shapes has mainly focused on geometry, with the aim of defining effective representations and accurate approximations of objects [FS97, Spa97, FS98], the new challenge in Shape Modelling is how to interpret and retrieve models from large repositories.

Shape analysis and understanding are basic tools to construct object descriptions, as the

---

processes aiming at detecting the main features of a given shape and their configuration.

Several methods have been proposed in the literature, ranging from segmentation methods to skeletonization techniques, to the computation of global shape signatures. These tools are fundamental for classification and retrieval, to create new shapes using innovative modeling by composition paradigms [FKS<sup>+</sup>04] and devise effective semantics-driven rendering of complex shapes. Beside a large number of contributions in image processing [BJ85, Pav95, DM98, Lon98], in Computer Graphics shape segmentation is now recognized to be a key ingredient in many shape manipulation processes [Sha06], and shape descriptors are crucial for 3D search and retrieval [VH01, TV04b, BKS<sup>+</sup>05, IJL<sup>+</sup>05b, DP06].

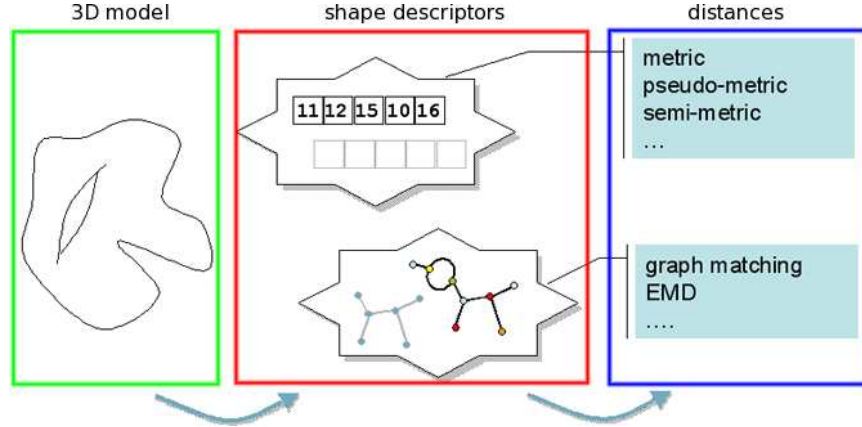
Shape interpretation is especially relevant for the perception of complex forms, in which the ability to vary the level of descriptive abstraction is the key to recognizing and classifying highly complex shapes. In particular, a model description should be *accessible/computable* and there should exist a clear definition of the object classes for which the model is suitable and the representation is unique, the so-called *scope/uniqueness*. In addition, to similar objects must correspond similar descriptions, even if the model should reflect subtle differences between shapes [NP85].

## Contribution

Among the different domains related to Computer Science, this work deals with the aspects of computational topology that are the most related to shape analysis and description. In this direction, the contribution of this thesis is both the analysis of existing approaches and the definition and use of new shape descriptors. Efforts have been devoted to systematize both the theoretical and descriptive aspects, providing an original interpretation of the shape description framework at the light of the complexity of the mathematical structure embedded in the descriptors.

Starting from the theoretical background provided by differential topology, and in particular Morse theory [Mil63], we derive a conceptual model for shape description based on topology and geometry suitable for shape abstraction. In our framework, we analyze different aspects of shape description and matching pursuing the problem of moving from a geometric model to a computational one. In particular, we propose a computational approach that is suitable to analyze any data that can be modeled as a subspace of  $\mathbb{R}^n$  and whose shape characteristics may be measured by one or a set of real functions. To fulfill this task, the first step is to associate a signature (i.e., a kind of code) to a geometric model. In our understanding, our description organizes the object shape in a way which reflects the topology of the original model. Moreover, it is suitable for those application tasks, like shape recognition, simplification, retrieval and similarity evaluation, for which the capability of detecting the main shape characteristics and evaluating how much two shapes overlap or have common components is fundamental.

---



**Figure 3:** Shape matching framework: one or more shape descriptors, or signatures, are associated to a 3D model; then a distance between shape descriptors is defined.

In general, the shape matching problem is solved/approached in the computational settings by instantiating a pipeline which is made of two basic steps as shown in Figure 3. First, the shape (usually in  $\mathbb{R}^3$ ) is mapped into an abstract space that contains descriptions, while the second phase deals with the definition/use of distances, or similarity/dissimilarity measures, between shape descriptors. In this scenario, our approach mainly deals with the definition of shape descriptors that are both concise, significant, computationally efficient and mathematically well-defined.

To this aim, we introduce the *Shape Graph (SG)* descriptor that combines the classical topological shape abstraction provided by the Reeb graph with additional geometric information able to code the local shape characteristics. The Shape Graph can be used to describe the overall appearance of the shape object, discarding irrelevant details and classifying its topological type. Two techniques to associate geometry information to a topological graph have been explored and applied to shape matching and retrieval context. Then, the Shape Graph has been extended to the comparison of sets of objects. In addition, we have adopted the Shape Graph to extend for the first time the size descriptor to surface models. the Shape Graph can be used to describe the overall appearance of the shape object, discarding irrelevant details and classifying its topological type. As further contribution, we also address the direct computation of the size function descriptor with respect to multivariate functions on data of any dimension (multi-dimensional size descriptor).

To prove the effectiveness of the shape description framework proposed, both descriptors (Shape Graph and multi-dimensional size descriptor) software prototypes have been implemented. Moreover, all tools have been validated in several application domains with main focus on shape matching applications.

## Overview

Before presenting a detailed layout of the chapters of this thesis, we would like to make some general remarks about their structure. After preliminary definitions in Chapter 1, Chapter 2 is mainly devoted to the presentation of tools based on real functions for shape analysis and synthesis. Our contribution to the definition of new descriptors is detailed in the Chapter 3 while their application to the computation context is described and discussed in Chapter 4. The preamble of each chapter will contain a short motivation of the topics proposed there, while a short summary at the end of the chapters 2, 3 and 4 will collect the main results, draw some conclusions and outline perspectives for possible future work. Finally, we remark that these chapters list a set of original publications related to the work proposed in each chapter. The reminder of this dissertation is organized as follows.

First of all, **Chapter 1** contains preliminary definitions and concepts related to the mathematical background and the discretization issues behind the shape descriptors discussed throughout the thesis. Then, the Chapter lists the real functions used in the 3D shape analysis.

In **Chapter 2** we overview methods for shape description, focusing on techniques that make use of theoretical frameworks that are developed mainly for *classes* of real functions. These methods describe a shape using both geometry and topology information: geometric, or metric, properties and attributes are crucial for characterizing specific instances of features, while topological properties are necessary to abstract and classify shapes according to invariant aspects. More in detail, we consider the Morse and Morse-Smale complex, the Reeb graph and the contour tree, the size functions, the persistent homology tools, the Morse descriptor and methods derived from the distance transform, such as the curve skeleton. All approaches are analyzed with respect to theory, computation and application providing comparative remarks and application domains.

**Chapter 3** details our contribution to shape description. Since we aim at characterizing a shape on the basis of basic functions that act as *lens* through which the shape is inspected; first, we overview some of the functions more popular for shape description. Starting from one or a set of real functions we derive two kinds of shape descriptors. First, we consider a Shape Graph that derives from a discretization of the Reeb graph definition [Ree46, BGSF08b] and couples the topological graph with a set of geometric attributes. In particular, we propose two possible ways of associating geometric attributes to a Reeb graph and also extend this graph representation to 3D scenes made of a set of objects. Second, we investigate the extension of the size function computation to 3D shape. In the initial phase, the size functions are extended to closed triangle meshes using the shape graph and geometric attributes as measuring functions; then their computation has extended to multi-variate functions defined both on surface and volume models.

Since the descriptors proposed rely on the use of real functions defined on the objects at

---

hands, it is important to compare the properties expressed by two or more different functions defined on the same shape. This is presented at the end of the Chapter, where we also describe how to build a new function whose characteristics are almost everywhere orthogonal to a given one, and therefore we contribute to the solution of finding a “basis” of functions that independently describe the shape properties.

**Chapter 4** focuses on the application domains of the descriptors proposed in the thesis. The first application we consider is shape matching and retrieval. In particular, we test our tools with respect to well known retrieval techniques. Then, we move from global similarity evaluation to partial matching and sub-part correspondence problems. We highlight that our Shape Graph is well suited for partial matching applications and we show its performance on the SHREC benchmark. A generalization of the partial correspondence problem is scene comparison, which is addressed in Section 4.1.3. Moreover, we address the shape classification problem, highlighting how shape prototypes generated on the basis of the Shape Graph may reduce the number of comparison operations involved in the classification process. Finally, the Chapter ends showing how the shape characterization process involved in Shape Graph extraction is also suited for the selection of the best view of an object.

Conclusions and hypotheses on further developments of the topics described throughout the thesis are discussed at the end of the dissertation (**Chapter 5**).





# Chapter 1

## Background notions

This Chapter outlines the background notions related to the theoretical aspects of the methods discussed in the thesis. Concepts and theories that are frequently encountered in the literature are briefly listed. More in detail, Section 1.1 provides a review of the mathematical concepts underlying the methods presented in this thesis, while Section 1.2 sketches some of the issues arising when applying concepts and theories defined in the continuum to a computational setting. Then, we end the Chapter with an overview of the real functions mainly used by the shape descriptors considered.

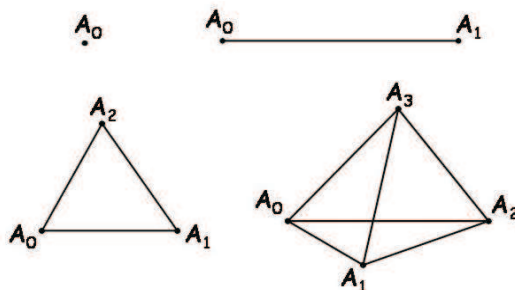
### 1.1 Mathematical background

*Topological spaces* are mathematical structures that allow the generalization of concepts such as closeness, limits, connectedness, or continuity, from the Euclidean space  $\mathbb{R}^n$  to arbitrary sets of points. This is achieved using relationships between sets, rather than distances between points. A detailed treatment of this subject can be found in [Wil70].

In order to construct topological spaces, one can take a collection of simple elements and glue them together in a structured way. Probably the most relevant example of this construction is given by *simplicial complexes*, whose building-blocks are called simplices.

A *k-simplex*  $\Delta^k$  in  $\mathbb{R}^n$ ,  $0 \leq k \leq n$ , is the convex hull of  $k + 1$  affinely independent points  $A_0, A_1, \dots, A_k$ , called *vertices*. Figure 1.1 shows the simplest examples of simplices:  $\Delta^0$  is a point,  $\Delta^1$  an interval,  $\Delta^2$  a triangle (including its interior),  $\Delta^3$  a tetrahedron (including its interior).

A *k-simplex* can be oriented by assigning an ordering to its vertices: two orderings of the vertices that differ by an even permutation determine one and the same orientation of the *k-simplex*. In this way, each *k-simplex* with  $k > 0$  can be given a positive or a negative orientation. The *oriented k-simplex* with ordered vertices  $(A_0, A_1, \dots, A_k)$  is de-



**Figure 1.1:** Examples of 0-, 1-, 2- and 3-simplices.

noted by  $[A_0, A_1, \dots, A_k]$ , whereas the  $k$ -simplex with opposite orientation is denoted by  $-[A_0, A_1, \dots, A_k]$ .

A *face* of a  $k$ -simplex  $\Delta^k$  is a simplex whose set of vertices is a non-empty subset of the set of vertices of  $\Delta^k$ . A *finite simplicial complex* can now be defined as a finite collection of simplices that meet only along a common face and their faces of any dimension. A concrete example of a simplicial complex is given by triangulated surfaces, where the vertices, edges and faces of the triangulation are 0-, 1- and 2-simplices, respectively. The *dimension* of a simplicial complex is the maximum dimension of its simplices. For more details on simplicial complexes we refer to [Mun00]. In addition, we mention that a more general class of spaces are *cell complexes* [GH81].

The approach adopted by *algebraic topology* is the translation of topological problems into an algebraic language, in order to solve them more easily. A typical case is the construction of algebraic structures to describe topological properties, which is the core of homology theory, one of the main tools of algebraic topology, see [Spa66]. In particular, the *homology* of a space is an algebraic object which reflects the topology of the space, in some sense counting the number of holes.

*Morse theory* can be seen as the investigation of the relation between functions defined on a manifold and the shape of the manifold itself. Intuitively, a *manifold* is a topological space that is locally Euclidean, meaning that around every point there is a neighborhood that is topologically the same as the open unit ball in  $\mathbb{R}^n$ ; the number  $n$  is the *dimension* of the manifold, [Hir97].

The key feature in Morse theory is that information on the topology of the manifold is derived from the information about the critical points of real functions defined on the manifold. Let us first introduce the definition of Morse function, and then state the main results provided by Morse theory for the topological analysis of smooth manifolds, such as surfaces. A basic reference for Morse theory is [Mil63], while details about notions of geometry and topology can be found, for example, in [Hir97].

Let  $M$  be a smooth compact  $n$ -dimensional manifold without boundary, and  $f : M \rightarrow \mathbb{R}$  a smooth function defined on it. Then, a point  $p$  of  $M$  is a *critical point* of  $f$  if we have

$$\frac{\partial f}{\partial x_1}(p) = 0, \frac{\partial f}{\partial x_2}(p) = 0, \dots, \frac{\partial f}{\partial x_n}(p) = 0,$$

with respect to a local coordinate system  $(x_1, \dots, x_n)$  about  $p$ . A real number is a *critical value* of  $f$  if it is the image of a critical point. Points (values) which are not critical are said to be *regular*. A critical point  $p$  is *non-degenerate* if the determinant of the *Hessian* matrix of  $f$  at  $p$

$$H_f(p) = \left( \frac{\partial^2 f}{\partial x_i \partial x_j}(p) \right)$$

is not zero; otherwise the critical point is *degenerate*.

We say that  $f : M \rightarrow \mathbb{R}$  is a *Morse function* if all its critical points are non-degenerate. The Morse Lemma states that the function  $f$  looks extremely simple near each non-degenerate critical point  $p$ . Indeed, we can choose appropriate local coordinates  $(x_1, \dots, x_n)$  around  $p$ , in such a way that  $f$  has a quadratic form representation:  $f(x_1, \dots, x_n) = f(p) - \sum_{i=1}^{\lambda(p)} x_i^2 + \sum_{i=\lambda(p)+1}^n x_i^2$ . Therefore, intuitively, the index of a critical point is the number of independent directions around the point in which the function decreases. For example, on a 2-manifold, the indices of minima, saddles, and maxima are 0, 1, and 2, respectively.

An important property is that a Morse function defined on a compact manifold admits only finitely many critical points, each of which is isolated. This means that, for each critical point  $p$ , it is always possible to find a neighborhood of  $p$  not containing other critical points.

Topological information about  $M$  is captured by the changes of the level sets and the lower level sets of  $M$  relative to the function  $f$ . The *level set* of  $f$  corresponding to the real value  $t$  is the set of points  $V_t = \{p \in M \mid f(p) = t\} = f^{-1}(t)$ ;  $t$  is called an *isovalue* of  $f$ . The *lower level set* is given by  $M_t = \{p \in M \mid f(p) \leq t\} = f^{-1}((-\infty, t])$ .

Morse theory states that the topology of  $M_t$  stays unchanged (formally, the homotopy type is preserved) as the parameter  $t$  goes through regular values of  $f$ , while changes occur when  $t$  passes through a critical value.

In order to study the changes in the level sets  $V_t = f^{-1}(t)$ , an approach to Morse theory based on the attaching of handles [Mil65], rather than cells, can be used, as in [Gra71] for the case of surfaces. When  $f$  is defined on a surface, if  $t$  is a regular value for  $f$  then  $V_t$ , if not empty, is the union of finitely many smooth circles. Moreover, if  $a, b$  are real numbers such that  $a < b$ , then

1. if the set  $f^{-1}([a, b])$  contains no critical points for  $f$ , then  $V_a$  and  $V_b$  are diffeomorphic;
2. if the set  $f^{-1}([a, b])$  contains only one critical point of index 0 for  $f$ , then  $V_b$  is the union of  $V_a$  with a circle;

3. if the set  $f^{-1}([a, b])$  contains only one critical point of index 2 for  $f$ , then  $V_b$  is diffeomorphic to  $V_a$  without one of its circles;
4. if the set  $f^{-1}([a, b])$  contains only one critical point of index 1 for  $f$ , then the number of connected components of  $V_b$  differs from that of  $V_a$  by  $-1$ ,  $0$  or  $1$  depending on the attaching map.

In case (4), the difference in the number of connected components is non-zero if the handle (in this case, the strip  $[0, 1] \times [0, 1]$ ) is attached without twists (or with an even number of twists), while it is 0 if there is an odd number of twists. The presence of an odd number of twists implies that the surface is non-orientable. Therefore, when the surface is embedded in  $\mathbb{R}^3$ ,  $V_a$  and  $V_b$  necessarily have a different number of connected components.

Morse theory asserts that changes in the topology of a manifold endowed with a Morse function occur in the presence of critical points; since most manifolds can be triangulated as simplicial complexes and a Morse function can be discretized on simplices, those changes in the topology can be interpreted in terms of homology, [GH81].

## 1.2 Computational background

Generally speaking, a *shape* can describe any phenomenon in the real or virtual world which is characterized by a geometric nature. Thus, we have shapes acquired from existing objects (e.g. images, 3D scans), shapes that are defined by sampling mathematical surfaces (e.g. implicit or algebraic surfaces), or shapes that are defined by the behavior of a physical quantity (e.g. temperature, or viscosity of a fluid).

From a mathematical point of view, a shape can be modeled as a topological space in  $\mathbb{R}^n$ , and usually we consider shapes in  $\mathbb{R}^3$  when dealing with applications in Computer Graphics. Often these shapes are abstracted as manifolds embedded in  $\mathbb{R}^n$ , usually orientable and smooth.

This first stage in the modeling pipeline is known as *mathematical modeling* [Req80] and consists of formulating the basic properties that an abstract computational model should have. In most of the methods discussed (e.g. contour trees, Reeb graphs, Morse decompositions), shapes are abstracted as manifolds, but we also discuss methods that assume the shape to be a more general topological space (e.g. in the context of size theory).

The next step is the selection of a computational representation scheme consistent with the mathematical model. *Cell decompositions* are the most common geometric model used in computer graphics and CAD/CAM [M88, Mor86]. From a historical perspective, the first type of model used was the *wireframe* model, which consists of the representation of edge curves and points on the object boundary. This has been developed further into *surface*

---

models, that provide the full representation of the geometry of the boundary of a 3D shape, and by *solid* models, that encode a shape as a composition of volumes.

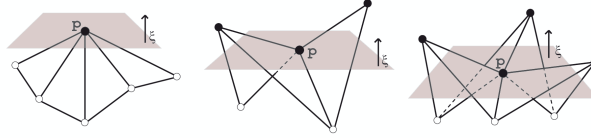
Shapes defined as graphs of scalar fields are the simplest kind of shapes studied in the methods surveyed. Formally, a *scalar field* is defined by any real-valued function  $f : \mathbb{R}^n \rightarrow \mathbb{R}$ . In practice,  $f$  is considered only on a domain  $D \subset \mathbb{R}^n$  corresponding to a  $d$ -dimensional interval in  $\mathbb{R}^n$ , where  $d \leq n$ . Generally, a  $d$ -dimensional scalar field  $\Gamma$  is denoted by the pair  $(D, f)$  and the values of  $f$  describe a physical phenomenon measured at a discrete set of points in  $D$ . Usually, such points form the vertices of a hypercubic grid.

Representing a scalar field  $\Gamma$  in a computational setting implies the discretization of the domain  $D$  as well as the discretization of the range of  $f$ . The domain is generally partitioned through a  $d$ -dimensional simplicial or cell complex. Thus, a model for a scalar field is called *simplicial* if the domain decomposition is a simplicial complex, while it is called *regular* if the domain is discretized through a *regular grid* or *digital spaces*, i.e., a cell complex in which all the cells are hypercubes. These cells are known as pixels in 2D and voxels in 3D, when the scalar field describes a two-dimensional or three-dimensional image [EL03, KR04]). In a regular model the value of the field can be associated either with vertices or with  $d$ -cells, and it is interpolated at other locations.

In general, most of the methods discussed in this dissertation deal with shape models represented by simplicial meshes (e.g. 3D shapes discretized as triangle or tetrahedral meshes, scalar fields defined on simplicial decompositions of the domain) or regular grids (e.g. scalar fields defined on rectangular 2D or 3D grids). Moreover, a binary image in 2D, 3D and higher dimensions is represented through a geometric realization of a graph that encodes the non-zero elements of the image as nodes and the neighborhood adjacency among the digital points as edges. Simplicial meshes are usually based on a piece-wise linear interpolation of the shape geometry. Regular grids define a step-wise or analytical approximation of the shape geometry, according to the type of interpolation associated with the hypercubes.

Finally, we mention that differential concepts have been adapted to discrete settings (e.g. the definitions of critical points proposed in [Ban67, Ban70, EM90]) or re-defined, like the Discrete Morse theory [For98, For02b]. Since in Section 3.3.2 we adopt the definition of critical points defined by Banchoff [Ban70], here we briefly summarize his approach.

The Banchoff's definition was originally devoted to height functions defined over polyhedral surfaces, or cell complexes, and are currently used by most of the computational approaches. This method uses a geometric characterization of the critical points that takes into account the position of the tangent plane with respect to the surface. A small neighborhood around a local maximum and minimum never intersects the tangent plane, while for saddles the small neighborhood is split into at least four pieces (see Figure 1.2). The number of intersections  $r$  is used to associate a *discrete index*  $i(p, f)$  to a critical point  $p$  with respect to a given  $f$ . Under the assumption that the function is general, i.e.  $f(v) \neq f(w)$  for all  $v$  and  $w$  vertices of  $D$ , critical points may occur only at those points  $p$  whose index  $i(p, f) = 1 - \frac{1}{2}r$  is not



**Figure 1.2:** Configuration of vertices around a maximum, saddle, and monkey saddle point.

null. In particular, the index is equal to 1 for maximum and minimum points, and can be a negative integer for saddles. For example, a monkey saddle has index  $-2$ . Finally, Banchoff proved that the relation  $\sum_{p \in D} i(p, f) = \chi(D)$ , where  $\chi$  denotes the Euclidean characteristic, holds for polyhedra.

### 1.3 Real functions

In the variety of real-valued functions that have been used in the Computer Vision and Graphics literature for characterizing relevant features of objects and shape matching applications, we summarize those that hold with our shape description framework and rely to the topics treated in the remainder of this thesis. Notice that, when dealing with applications in the discrete context, also functions that are not Morse have been considered.

**Height and elevation functions** The height [SKK91, FK97] function is among the most intuitive and simple choices for analysing the shape of an object; since it depends on the direction considered, its usage is preferred for applications in which objects have a natural predefined direction (Figure 1.3(a)).

A more elaborate characterization of the shape according to differences in the elevation value is provided by the *elevation* [AEHW06] function, which derives from the traditional height function but aims at a rotation invariant analysis. The notion of elevation captured by this function measures how much a point is relevant in its normal direction with respect to it of the height function in all directions.

**Distance functions** To describe a shape  $S$ , the Euclidean distance from a point  $p \in \mathbb{R}^3$ ,  $f^1(v) = \|v - p\|_E, \forall v \in S$ , [FK97, SV01] has also been used. In case  $p$  is the center of mass (barycenter) of  $S$  (see Figure 1.3(b)),  $f$  is invariant to rotations with respect to the point  $p$  and detects protrusions (resp. hollows) of  $S$  with respect to  $p$  as regions of influence of maxima (resp. minima) of the function  $f$ .

**Geodesic-based functions** Since shape properties can be effectively characterized by measuring distances between feature points or by evaluating the elongation of the shape,

the approaches based on the geodesic distance generally provide an isometry invariant characterization of a shape [SSK<sup>+</sup>05, BBK06]. Geodesic distances from selected feature points have been evaluated on mesh vertices [MP02, EK03], as well as average geodesic distances [HSKK01], (Figure 1.3(a)). In the latter case the function  $f$  may be expressed as  $f(v) = \int_{p \in S} g(v, p) dS$ , where  $g(v, p)$  represents the geodesic distance between  $v$  and  $p$ , when  $p$  varies on  $S$ . Variations of this function can be found in [ZMT05, KT03, GSCO07].

**Curvature-based functions** Curvature-based analysis measures the local concavity/convexity of a shape. A classical method to evaluate the curvature in a vertex  $v$  is the standard angle-deficit approximation, as in [PKS<sup>+</sup>03]:  $C(v) = \frac{2\pi - \sum_i \theta_i}{3 \sum_i A_i}$  where  $\theta_i$  and  $A_i$  are the apex angles and areas in the one ring of triangles incident on  $v$ , respectively. Unfortunately, the straightforward evaluation of curvature analysis on mesh vertices is rather sensitive to noise or small features and to the quality of the shape discretization in terms of sampling density and tiny triangles [Spa97].

More robust computation is achieved either using variations of the curvature evaluation function (e.g. [GCO06]), polynomial surface fitting [ZP01], or with a multi-scale curvature evaluation based on the ratio between the length of the intersection line of a collection of spheres centred into mesh vertices and their radius [MPS<sup>+</sup>04].

**Harmonic and Laplacian-based functions** The harmonic [NGH04, Flo97, PP93] and Laplacian-based functions [RWP06, DBG<sup>+</sup>06] have been recently introduced into Computer Graphics literature to provide a set of descriptors that are intrinsic to the shape, as they are a discretization of the Laplacian operator:

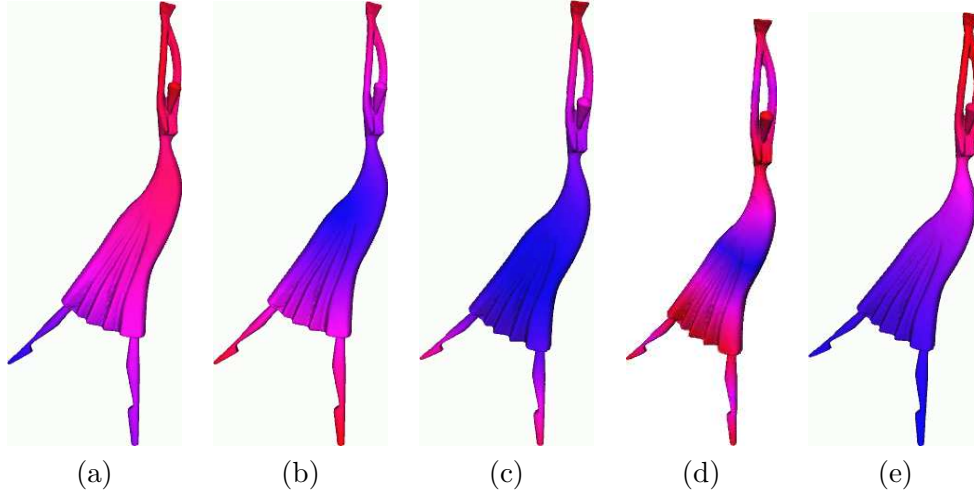
$$\Delta f = \sum_{i=1}^n \frac{\partial^2 f}{(\partial_i x)^2} \quad (1.1)$$

A solution of the Laplacian eigenvalue problem:

$$\Delta f = -\lambda f. \quad (1.2)$$

is named an *eigenfunction* of the shape and, in the planar 2D case, it can be understood as the natural vibration form of a homogeneous membrane with the *eigenvalue*  $\lambda$ . Several discrete methods exist to solve the Laplacian eigenfunction problem, one of the most popular are the *Finite Element Method* [RWP06]. Figure 1.3(d) shows the second eigenfunction of the dancer model.

**Distance transform function** Finally, the *distance transform* is a well known function in the Computer Vision and Graphics literature, that is related to the medial axis radius.



**Figure 1.3:** (a) Height function, (b) Euclidean distance from the barycenter, (c) average geodesic distance in [HSKK01], (d) distance from curvature extrema, (e) first eigenfunction of the Laplacian matrix of the model.

For a 3D shape, this means that the distance transform measures its interior volume rather than its surface. In any dimension, a distance function  $f : \mathbb{R}^n \rightarrow \mathbb{R}$  is defined over the space  $\mathbb{R}^n$  where the shape  $S$  is embedded by the relation [DGG03]:

$$f(x) = \min_{p \in S} \|p - x\|^2, \forall x \in \mathbb{R}^n.$$

Note that the distance transform function is not a Morse function.

Another function that provides a volumetric rather than a boundary characterization is the *local diameters* function [GSCO07]. Differently from the distance transform it aims at measuring the shape by computing the *diameter* of the volume enclosed by the surface.

In general, it is difficult for a single function to fully describe the properties of a complex shape. Moreover, we conjecture that the number of functions necessary to describe a shape depends on its complexity rather than on the storage size. This is the reason why we are focusing on shape descriptors characterized by the flexibility with respect to the function  $f$ ; in particular, we have approached the problem of considering more functions at the same time, see Section 3.2.2.



## Chapter 2

# Describing shapes using real functions

The use of structures for shape description has been widely addressed in Computer Graphics and Vision. In this Chapter we focus on descriptors grounded in Morse theory that make use of the properties of real functions.

The intuition behind Morse theory is that of combining the topological exploration of a shape  $S$  with quantitative measurements of its geometrical properties provided by a mapping function  $f$ , defined on  $S$  [Mil63]. Integrating the classifying power of topology with the differentiating power of geometry enables us to extract information about shapes at different levels, taking into account global as well as local shape properties. Therefore, we focus on methods for shape understanding that find their roots in Morse theory, ranging from simple to complex shapes, from single to arbitrary mapping functions, and from unstructured to algebraically structured sets of descriptors [BFD<sup>+</sup>08].

Broadly speaking, the methods discussed in the thesis can be divided into four groups: methods studying the configuration of critical points on the shape boundary (Morse and Morse-Smale complexes), methods studying the evolution of the level sets of  $f$  (contour trees and Reeb graphs), methods studying the evolution, or growth, of the lower level sets of  $f$  (size theory, persistent homology and Morse shape descriptors) and methods that refer to distance transform and, more in general, to skeletal structures. The first three groups of methods discussed reflect, to different extents, the modularity of the approaches based on Morse theory, especially from the point of view of the applications they have been traditionally designed for while the fourth is more oriented to a practical definition of the skeleton.

Morse and Morse-Smale complexes were introduced in Computer Graphics for the analysis of two-dimensional scalar fields, but recently their use has also been extended to handle generic 3D shapes. These structures provide a view of shape properties from the perspective of the *gradient* of the mapping function [BDFP07]. Intuitively, the aim is to describe the shape

by decomposing it into cells of uniform behavior of the gradient flow. The decomposition can be interpreted as having been obtained by a network on surface, that joins the critical points of the mapping function  $f$  through lines of steepest ascent or descent of the gradient.

The theory behind Morse and Morse-Smale complexes is of general application, and is also related to the theory of dynamical systems. These two views are clearly reflected in the literature and a considerable number of techniques have been developed to extract critical points and lines, especially for terrain surface modelling and analysis.

Contour trees have been used mainly to study the shape of scalar fields, and no distinction is made between the shape and the function used to analyze it: both coincide with the scalar field itself. Contour trees describe the shape of a scalar field  $f$  by analyzing the evolution of its level sets, as  $f$  spans the range of its possible values: components of level sets may appear, disappear, join, split, touch the boundary or change genus. The contour tree stores this evolution and provides a compact description of the properties and structure of the scalar field. Contour trees, however, could, in principle, be defined for any shape with any mapping function, and the theory behind them is general. Contour trees, in all their variants, are discussed with emphasis on the methods developed in Computer Graphics and with pointers to similar structures defined in Computer Vision.

The generalization of a contour tree is given by Reeb graphs, even if their definition is slightly different, as presented in the literature [BGSF08b]. While the definition and use of contour trees developed mainly as an answer to computational issues, Reeb graphs have a more theoretical nature. Their definition and theoretical study date back to 1946, thanks to the research work of a French mathematician, George Reeb. With respect to the modularity of Morse theory, Reeb graphs are the first example of a fully modular framework for studying the shape of a manifold: here the shape exists by itself and the function used to study it can be arbitrarily chosen. In recent years, Reeb graphs have become popular in Computer Graphics as a tool for studying shapes through the evolution and arrangement of the level sets of a real function defined over the shape. Reeb graphs effectively code the shape, both from a topological and geometrical perspective. While the topology is described by the connectivity of the graph, the geometry can be coded in a variety of different ways, according to the type of applications the Reeb graph is devised for.

Contour trees and Reeb graphs are frequently associated, in the literature, with the concept of skeletal graphs or centerline skeletons. By coding the centroids of each level set, it is indeed very easy to trace a kind of centerline spanning the volume enclosed by the shape. Centerline skeletons are very popular in Computer Graphics and Vision, and are, in principle, related to the medial axis transformation, in the sense that they represent an effective way of reducing a complex 3D shape to a simple one-dimensional geometric abstraction [CSM05, LLS92]. Probably the medial axis transformation is the best-known structure and provides a decomposition of the shape in protrusions detected by spheres of different radius inscribed in the shape, [Blu67]. While the notion of centerline skeleton and medial axis, per

se, does not fall within the ambit of this overview, we discuss some recent results in that specific field in the section devoted to Reeb graphs, in order to point out their similarities in theory and applications.

Besides the possibility of adopting different functions for describing shapes, at a higher level of abstraction, the modularity of the approach based on Morse theory can be extended to the choice of the space used to represent the shape, or phenomenon, under study. The third group of methods reflects this higher degree of modularity, and it is concerned with methods allowing one or more real functions to be defined on spaces associated with the shapes under study. Size theory and persistent homology theory fall in this last group and are characterized by the possibility of varying the space underlying the shape and the real functions defined on it. Furthermore, an extensive use of algebraic structure characterizes these techniques.

Size theory has been developed since the beginning of the 1990s with the idea of defining a suitable mathematical setting for the problem of shape comparison, and, as such, it relies on four basic concepts: the natural pseudo-distance between size pairs, as a key tool for shape comparison, and the size functions, the size homotopy groups and the size functor for shape description and discrimination. A common property shared by these descriptors is that shapes are studied by varying the underlying space and the real functions defined on it.

Persistent homology follows a similar approach, but introduces another paradigm, *persistence*, which is based on growing a space (i.e., the shape) incrementally, and analyzing the topological changes that occur during this growth. The occurrence and placement of topological events (e.g., creation, merging, cancellation of the connected components of the lower level sets) within the history of this growth characterize the shape. Persistent homology aims to define a scale of the *relevance* of these topological events, characterizing the features of the shape. The main assumption is that longevity is equivalent to significance. In other words, a significant topological attribute must have a long life-time in a growing complex.

Also, another contribution pertaining to this class is discussed, the Morse shape descriptor, which differs from the other two, as it makes use of the theory of relative homology groups to define a shape description.

The medial axis nicely simulates the human intuition and it is well-suited for shape matching especially for 2D shapes, like for example done in [SKK04] using shock graphs, and more recently proposed for 3D shapes through thinning [SSGD03, SBTZ02, CSM05, ZSM<sup>+</sup>05, IJL<sup>+</sup>05a]. Several geometric descriptors have been proposed for associating to the nodes of a skeletal graph the description of the related model sub-parts [BAB<sup>+</sup>07]. A minimal solution consists in coding in a vector the *relevance* of the skeletal edges incident in a node (e.g. edge length, diameters and average circumference of the skeleton loops) as proposed in [SSGD03, IJL<sup>+</sup>05a]. Another strategy is to use geometric descriptors able to support global comparison of 3D shapes, like the mean curvature histogram [ZSM<sup>+</sup>05], or associating a

weight (for example the volume) to the centroids of a shape segmentation, like proposed in [DGG03].

## 2.1 Morphology-based shape descriptors

The intuition behind Morse and Morse-Smale complexes is nicely described by Maxwell [Max70]:

Hence each point of the earth's surface has a line of slope, which begins at a certain summit and ends in a certain bottom. Districts whose lines of slope run to the same bottom are called basins or dales. Those whose lines of slope come from the same summit may be called, for want a better name, hills. Hence, the whole earth may be naturally divided into basins or dales, and also, by an independent division, into hills, each point of the surface belonging to a certain dale and also to a certain hill.

If we consider the height function on a terrain, the partition of the surface into its hills corresponds to the decomposition defined by the unstable, or ascending, Morse complex. Similarly, the decomposition of the surface into its dales corresponds to the partition defined by the stable, or descending, Morse complex. If we overlap the decompositions based on the hills and on the dales, we obtain what is called a Morse-Smale decomposition.

This intuitive notion generalizes to any smooth surface and any mapping function  $f$ . The distinctive characteristics of Morse and Morse-Smale complexes are that they provide the study of shape properties from the perspective of the *gradient* of the mapping function. Morse and Morse-Smale complexes describe the shape by decomposing it into cells of uniform behavior of the gradient flow and by encoding the adjacencies among these cells in a complex which describes both the topology and the geometry of the gradient of  $f$ .

The use of Morse and Morse-Smale complexes was originally introduced in Computer Graphics for the analysis of two-dimensional scalar fields, but, recently, their use has been extended to handle three-dimensional scalar fields as well as generic 3D shapes. The theory behind Morse and Morse-Smale complexes, however, is of general application and has its roots in the theory of dynamical systems [PM82]. Moreover, Morse and Morse-Smale complexes are strongly related to visualization of vector field topology [HH89, TRW07]. Morse complexes correspond to a decomposition of the shape into either the stable or unstable manifolds associated with the mapping function, and the Morse-Smale complex is defined as the intersection of the stable and unstable manifolds, under some hypotheses that will be discussed below. Alternatively, this decomposition can be interpreted as having been obtained by joining the critical points of the mapping function  $f$  by lines, in the case of a two-dimensional

---

scalar field, (or surfaces in the case of a three-dimensional scalar field), of steepest ascent or descent of the gradient.

These two views are clearly reflected in the literature. A considerable number of algorithms have been developed for extracting critical points and lines, with a specific focus on terrain modeling and analysis. Morse and Morse-Smale complexes have been extensively studied, mainly for the understanding and visualization of scalar fields, but also for more general applications in shape analysis, by using as a mapping function the curvature [MW99, Pag03], or the Connolly function [CCL03].

In this section, we review relevant works reported in the literature that cover both interpretations of Morse and Morse-Smale complexes, including also methods that are oriented towards a segmentation of the shape into catchment basins of its minima, that again can be seen as a geometric interpretation of the same mathematical concept, namely regions of uniform flow of the gradient vector field of the mapping function. In other words, we have collected all methods that are rooted, explicitly or not, in the same mathematical framework. We have interpreted different classes of methods as different computational approaches for detecting the same geometric and topological characterization of a shape.

### 2.1.1 Morse complex

The definition of Morse complexes relies on the concepts of critical point and of integral line, as discussed below.

#### 2.1.1.1 Theoretical aspects

Let  $M$  be a smooth compact  $n$ -manifold without boundary, and let  $f : M \rightarrow \mathbb{R}$  be a smooth Morse function. Let us also assume that  $M$  is embedded in  $\mathbb{R}^n$  or that a Riemannian metric is defined on  $M$ . An *integral line*  $\gamma : \mathbb{R} \rightarrow M$  of  $f$  is defined as a maximal path on  $M$  whose velocity vectors, or tangent vectors, agree with the gradient of  $f$ , meaning that  $\frac{\partial \gamma}{\partial s} = \nabla f(\gamma(s))$  for all  $s$  in  $\mathbb{R}$ . Each integral line is open at both ends, having its origin (i.e.,  $\lim_{s \rightarrow -\infty} \gamma(s)$ ) and its destination (i.e.,  $\lim_{s \rightarrow +\infty} \gamma(s)$ ) at critical points of  $f$  [PM82]. Note that the critical points are images of constant integral lines by themselves.

It can be shown that integral lines are pair-wise disjoint, that is, if their images share a point, then they are the same line. The images of integral lines cover the whole  $M$ , but if we consider the integral lines associated with the critical points of  $f$ , their images define a partition of  $M$ .

This partition is used to decompose  $M$  into regions of uniform flow, thus capturing the characteristics of the gradient field. More precisely, the *descending manifold* of a critical point  $p$  is the set  $D(p)$  of points that flow towards  $p$ , and the *ascending manifold* of  $p$  is the

set  $A(p)$  of points that originate from  $p$ . In formulae:

$$A(p) = \{q \in M : \lim_{t \rightarrow +\infty} \gamma_q(t) = p\}$$

$$D(p) = \{q \in M : \lim_{t \rightarrow -\infty} \gamma_q(t) = p\},$$

where  $\gamma_q$  is the integral line at the point  $q$ . Note that the descending manifold of  $f$  is the ascending manifold of  $-f$ .

In the mathematical literature, the term *unstable* is used instead of ascending, and the term *stable* is used instead of descending [PM82]. Note also that the partition into ascending manifolds is similar to watershed decomposition [Mey94, VS91, MW99], as we will discuss below.

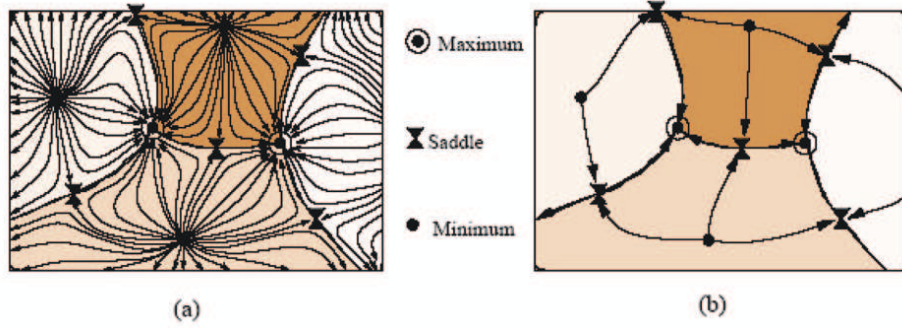
Here, we follow the terminology and notations adopted in the description of the majority of the methods we discuss. Note that there are slight differences in the definitions of ascending or descending manifolds, depending on whether the critical points are considered as belonging to the manifolds or not. For instance, in [EHZ03, BEHP04], the critical points are added to the related descending and ascending manifolds.

The descending manifold of a critical point  $p$  of index  $i$  is an open  $i$ -cell. Similarly, the ascending manifold of a critical point of index  $i$  is a  $n - i$  open cell. For example, if  $M$  is a 2-manifold the descending manifold of a maximum is an open disk, that of a saddle is an open interval, and that of a minimum is the minimum itself.

The collection of all descending manifolds form a complex, called the *descending Morse complex*, and the collection of all ascending manifolds also form a complex, called the *ascending Morse complex*, which is dual with respect to the descending complex. For instance, when  $M$  is a 2-manifold, the 2-cells of the descending Morse 2-complex correspond to the maxima of  $f$ , the 1-cells to the saddle points, and the 0-cells to the minima. Symmetrically, the 2-cells of the ascending Morse 2-complex correspond to the minima of  $f$ , the 1-cells again to the saddle points, and the 0-cells to the maxima. An example of a decomposition of the domain of a two-dimensional scalar field into an ascending Morse complex is shown in Figure 2.1 (a). When  $M$  is a 3-manifold, the 3-cells of a descending Morse 3-complex correspond to the maxima, the 2-cells to the 2-saddles, the 1-cells to the 1-saddles, and the 0-cells to the minima. Symmetrically, the 3-cells of the ascending Morse 3-complex correspond to the minima, the 2-cells to the 1-saddles, the 1-cells to the 2-saddles, and the 0-cells to the maxima.

### 2.1.1.2 Computational aspects

The majority of the algorithms for Morse complex computation have been developed for two-dimensional scalar fields, or for scalar functions defined over a 2-manifold without boundary.



**Figure 2.1:** (a) The ascending Morse complex of a two-dimensional scalar field (the 2-cells correspond to the minima). (b) The Morse-Smale complex. Its 1-skeleton (the set of simplices of dimension 0 and 1) is the critical net.

Most of these algorithms use what we call a *boundary-based* approach, since they extract the complex, by computing the critical points and then tracing the integral lines joining them, or their approximations, starting from saddle points and converging to minima and maxima. Other algorithms use a *region-based* approach in the sense that they compute an approximation of the ascending and/or descending Morse complex by growing the 2-cells corresponding to the minima, or to the maxima, of the Morse function  $f$  defined on a manifold  $M$ .

In [DDM03b, MDD<sup>+</sup>07], algorithms have been presented for computing the descending and ascending Morse complexes for a 2D simplicial model. These algorithms have been applied to the segmentation and morphological analysis of terrain models, and the algorithm in [MDD<sup>+</sup>07] has also been applied to 3D shape segmentation. In both algorithms, the ascending and descending complexes are computed independently by applying a region-growing technique on the triangles of the simplicial model. Basically, the algorithms perform a breadth-first traversal of the dual graph of the triangle mesh, in which the nodes correspond to triangles of the mesh, and the arcs to the edges shared by edge-adjacent triangles. When extracting the descending Morse complex, maxima are extracted. A descending 2-cell  $C$ , associated with a current maximum  $m$ , is initialized with all the triangles in the star of  $m$ , which have not yet been assigned to any 2-cell. Then, the cell  $C$  is grown in a breadth-first fashion by adding one triangle at a time according to a criterion which is specific for each algorithm.

The algorithm described in [DDM03b] has been defined in a dimension-independent way and implemented for 3D simplicial models as well. In [MD07] the authors have defined the discrete gradient vector field associated with the decomposition produced by the algorithm in [DDM03b] and have shown that it is a subfield of the gradient field of a Forman function  $F$  whose restriction over the vertices of the simplicial model coincides with the given scalar field function  $f$ .

A region-based combinatorial algorithm for computing ascending manifolds has been proposed in [DGG03] to segment a 3D shape  $M$  into meaningful features. In this case, the shape is defined by a set of points belonging to its boundary and is discretized as a Delaunay tetrahedral mesh  $\Sigma$  with vertices at the data points. In this case, a distance function is defined over the space and, most importantly, the results are discussed for that specific function selection. Note that the distance function is not a Morse function, so we cannot properly talk about a Morse complex, but the authors are interested in the computation of the ascending manifolds of maxima. Maxima of the distance function are detected at a subset of the vertices of the dual Voronoi diagram of the mesh  $\Sigma$ . Approximations of the three-dimensional ascending manifolds are computed by a region-growing approach starting from the tetrahedra which contain the maxima. A discrete gradient field is computed by defining a flow relation between face-adjacent tetrahedra, whose transitive closure is acyclic [EFL98, GJ03].

In [NGH04], a boundary-based approach to the computation of the descending Morse complex is proposed. The algorithm is applied to a triangle mesh  $\Sigma$  discretizing a 2-manifold without boundary endowed with a Morse function  $f$  which minimizes the number of critical points. The algorithm extracts first the critical points and successively traces the descending 1-manifolds starting from the saddles, in no specific order. If a point  $p$  is a multiple saddle of multiplicity  $m$ , then  $m$  manifolds start at  $p$ , each corresponding to one connected component of the lower link of  $p$ . Descending 1-manifolds are constructed by moving along the edges of the triangle mesh by choosing each time a point with lowest height in the lower link of the current point (as in [BS98, TIS<sup>+</sup>95]). These manifolds can merge, but they cannot cross, and they do not separate after merging. Descending 1-manifolds are not allowed to pass through saddles other than their starting saddle. In other words, if a descending 1-manifold has reached a point  $p$  in the link of a saddle  $s$ , then the edge  $ps$  cannot be used to extend the manifold, i.e., saddle  $s$  is not taken into account when searching the neighbor of  $p$  with lowest height. Further improvements are proposed, which include handling flat regions, or plateaus, or surfaces with boundary. In the case of plateaus, each simply-connected flat region is considered as a single vertex. This is equivalent to collapsing edges connecting vertices in the flat region. Surfaces with boundary are treated by adding a vertex  $p$  for each boundary component, together with edges and triangles connecting  $p$  to vertices on the boundary loop, producing a model without boundary.

The ascending and descending Morse complexes can also be computed by applying the *discrete watershed transform*. The watershed transform was first introduced in image analysis for the segmentation of gray-scale images and several definitions exist in the discrete case [BL79, BM98, MR96, Mey94, VS91]. It provides a decomposition of the domain of a  $C^2$  function  $f$  into regions of influence of the minima, called *catchment basins*. The boundary of the catchment basins of the minima form the *watershed lines*. Catchment basins and watershed lines are described in terms of topographic distance, using the formalization proposed in [Mey94]. In the 2D case, if  $f$  is a Morse function, it can be seen that the catchment



basins of the minima of  $f$  are the 2-cells in the ascending Morse complex of  $f$  and the watershed lines are 1-cells in such complex. Through a change in the sign of function  $f$ , the descending manifolds of the maxima can be extracted, and thus, we can obtain the descending Morse complex for the original function. Several algorithms have been developed for the computation of the watershed transform (see [RM00, NC03] for a survey).

### 2.1.2 Morse-Smale complex

Morse-Smale complexes are defined for functions belonging to the important class of dynamical systems, called *Morse-Smale* systems. They are structurally stable on compact manifolds meaning that their structure is preserved under topological equivalencies of the manifold. Intuitively, this means that the topological behavior of the images of the integral lines does not change under small perturbations of the vector field [PM82]. This property is guaranteed when function  $f$  is a *Morse-Smale* function, that is, the descending and ascending Morse complexes intersect only transversally<sup>1</sup>. In 2D this means that, if an ascending 1-manifold intersects a descending 1-manifold transversally, they cross at exactly one point. An example can be found in Figure 2.1 (b), where the two lines in the 1-skeleton intersect at most in one point (a saddle). The importance of the above condition is that it is a stable and generic condition, which is independent of small perturbations of function  $f$  and of manifold  $M$ . In Figure 2.2, an example of function whose gradient is not Morse-Smale is shown. The height function on the torus defines four critical points, one maximum, one minimum and two saddles. The integral lines emanating from the higher saddle have the lower saddle as destination in (a), but a slight perturbation of the height directions causes the gradient to flow towards the minimum  $m$  instead, as illustrated in (b,c).

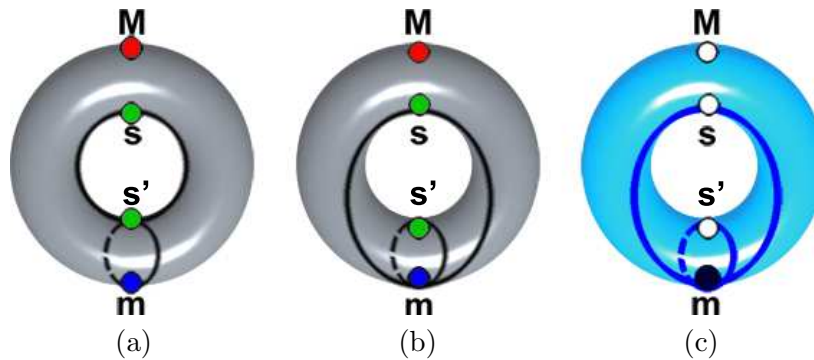
#### 2.1.2.1 Theoretical aspects

In the case of Morse-Smale functions, it is possible to define a complex, called the *Morse-Smale complex*, as the intersection of the ascending and descending manifolds. The cells of the *Morse-Smale complex* are the components of sets  $D(p) \cap A(q)$ , for all critical points  $p$  and  $q$  of function  $f$  [EHZ01, EHNP03]. Each cell of the Morse-Smale complex is the union of the integral lines sharing the same origin  $p$  of index  $i$  and the same destination  $q$  of index  $j$ . The dimension of the cell is given by the difference of the indices. Figure 2.1 (b) shows an example of a Morse-Smale complex. Notice that, in general, the closure of the cells of a Morse-Smale complex may not be homeomorphic to a closed ball.

The Morse-Smale complex is characterized by cells with a regular connectivity. In the 2D case, each saddle point  $p$  has four incident 1-cells, two joining  $p$  to maxima, and two joining  $p$  to minima. Such 1-cells alternate in a cyclic order around  $p$ . Also, the 2-cells are

---

<sup>1</sup>By definition, two submanifolds  $A$  and  $B$  of a manifold  $M$  intersect transversally in  $p$  if  $T_p A + T_p B = T_p M$  where  $T_p$  is the tangent space at  $p$ .

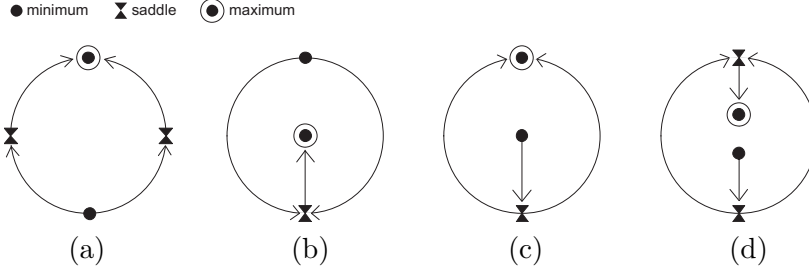


**Figure 2.2:** (a) The integral lines emanating from the higher saddle  $s$  reach  $s'$ , but a slight perturbation of the height direction causes the integral lines to reach  $m$  instead (b). The Morse complex is shown in (c). In (c) the black vertex correspond to the 0-cell, the blue lines represent the 1-cells while the light blue region is the 2-cell.

quadrangles whose vertices are critical points of  $f$  of index 1, 0, 1, 2 (i.e., saddle, minimum, saddle, maximum) in this order. In the 3D case, all 2-cells are quadrangles whose vertices are a minimum, 1-saddle, 2-saddle, 1-saddle in this order (*quadrangles of type 1*), or a 1-saddle, a 2-saddle, a maximum, a 1-saddle in this order (*quadrangles of type 2*). A 1-cell connecting a 1-saddle and a 2-saddle is on the boundary of four quadrangles that alternate between quadrangles of type 1 and type 2. The 3-cells are called *crystals* and are bounded by quadrangles [EHZ01, EHNP03].

It is interesting to note the similarity between the Morse-Smale complex and the configuration of slope districts defined in [Nac84], where the function is not necessarily a Morse-Smale one, but simply a Morse function. In particular, Nackman defined a graph, called the *Critical Point Configuration Graph (CPCG)*, in which the nodes represent critical points and the arcs represent the integral lines connecting them. The *CPCG* is a planar graph and its embedding on the domain  $M$  of  $f$  induces a partition of  $M$  into two-dimensional regions, called *slope districts*, characterized by the uniformity of the gradient flow. Since  $f$  is not necessarily a Morse-Smale function, there are configurations of the critical points of  $f$  which do not occur for Morse-Smale functions. Nackman shows [Nac84] that there are only four basic possible configurations for a slope district and they can be obtained, up to equivalence, by inserting saddle points in the arcs. These configurations are illustrated in Figure 2.3. The configurations in Figures 2.3(a), (b) and (c) are formed by saddle, minimum, saddle, and maximum. All three illustrate the possible types of 2-cells in a Morse-Smale complex. The ones in Figures 2.3 (b) and (c) correspond to degenerate situations, usually called *strangulations* [GNP<sup>+</sup>05]. The configuration in Figure 2.3(d) cannot happen for a Morse-Smale function, since ascending and descending 1-manifolds do not intersect transversally, but coincide.

The 1-skeleton of a Morse-Smale complex is a 1-complex formed by integral lines joining critical points. Similar structures among critical points have been widely studied in the



**Figure 2.3:** The four possible configurations for the slope districts in a CPCG.

literature under the name of *critical nets*. A graph representation of the critical net in a two-dimensional Morse-Smale complex is the so-called *surface network* [Pfa76, SW04], widely used in spatial data processing for morphological terrain modeling and analysis (see [Ran04] for an interesting collection of contributions on this specific topic).

### 2.1.2.2 Computational aspects

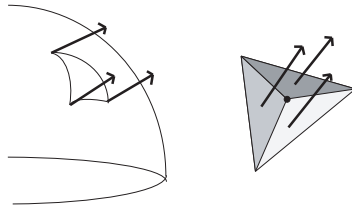
In general region-based methods aim to extract a Morse complex, while boundary-based approaches typically focus on the extraction of a Morse-Smale complex. In particular, when  $f$  is a Morse-Smale function, the Morse-Smale complex can be obtained as the intersection of the ascending and descending Morse complexes.

Most boundary-based methods for computing a Morse-Smale complex [TIS<sup>+</sup>95, BS98, EHZ01, BEHP03, Pas04, BP07] from two-dimensional complexes follow the same algorithmic approach, consisting of two basic steps:

- extract the critical points and unfold multiple saddles;
- compute the 1-cells of the Morse-Smale complex, or their approximations, by starting from the saddle points, and tracing two paths on the underlying shape model which stop at minima and maxima, respectively.

The extraction of the critical points is usually performed based on techniques implementing the classification by Banchoff [Ban67]. In [BS98], a different approach is used even if, as in the previous algorithms, the technique performs the classification of vertices with respect to the local neighborhood. In this case, at a given vertex of the simplicial model, the authors estimate the gradient as it can take a range of values based on the normals of the triangles incident in that vertex. As a result, critical points are defined as vertices at which the normal space of the incident triangles includes the vector  $(0, 0, 1)$  (see Figure 2.4).

The main difference among the different methods relies in the technique used to trace the integral lines that define the 1-cells: either the steepest ascent or descent is traced, or



**Figure 2.4:** The normal space for a cluster of triangles incident on a vertex.

approximated integral lines are used instead, provided that they respect the connectivity of the Morse-Smale complex.

The algorithms in [TIS<sup>+</sup>95, BS98, BP07] extract the integral lines forming the Morse-Smale complex by computing paths only along the edges of the triangle mesh, selecting the vertex of highest, or lowest, height at each step. As observed by [CLLR05], the time complexity of [TIS<sup>+</sup>95] is  $O(nc)$ , where  $c$  denotes the number of critical points. As a pre-processing step, to simulate that the critical points are non degenerate and there exist no boundary saddles, [BP07] adopted a symbolic perturbation of the mesh before extracting the Morse-Smale complex. The algorithms in [BEHP03, Pas04] estimate the gradient along 1-simplices and 2-simplices, and compute the ascending and descending paths not only along the edges, but possibly crossing triangles in order to follow the actual paths of steepest ascent, or descent.

In [EHZ01, EHNP03] the notion of *Quasi Morse-Smale (QMS)* complex is introduced as an intermediate step towards the computation of the Morse-Smale complex. The QMS is defined for 2D and 3D simplicial complexes, which triangulate a 2-manifold or a 3-manifold without boundary, respectively. The QMS has the same combinatorial structure of a Morse-Smale complex, but it differs from it in that the 1-cells in 2D, and the 1-cells and 2-cells in 3D are not necessarily those of maximal ascent, or descent. The idea behind a QMS, called *simulation of differentiability* is that of extending the smooth notions to the piece-wise linear case so as to guarantee that the complex has the same structural form of the smooth counterpart, and to achieve numerical accuracy via local transformations that preserve the structure of the complex [EHZ01].

The QMS is a splittable quadrangulation of  $M$  whose vertices are the critical points of  $f$  and whose arcs are strictly monotonic in  $f$ . The 0-cells of a QMS complex are the critical points of  $f$ , the 1-cells connect minima to saddles (1-saddles in 3D), maxima to saddles (2-saddles in 3D) and, in the 3D case, 1-saddles to 2-saddles [EHNP03]. Since the computed lines are an approximation of the integral lines in the smooth case, the algorithm resolves problems arising when merging and forking of paths occur. Once the QMS complex is computed, a series of operations, called *handle slides*, are applied to turn the QMS into a Morse-Smale complex. For 2-manifolds, it is possible to find a sequence of handle slides that brings the QMS close to a Morse-Smale complex, while how to construct a sequence of operations that yield the same result for 3-manifolds is still an open question [EHNP03].

The boundary-based algorithm in [EHNP03] extracts the Quasi Morse-Smale complex for a simplicial model of a three-dimensional scalar field by computing first the critical points through the reduced Betti numbers of the lower link of the vertices. Second, the descending Morse complex is computed and, finally, the algorithm extracts the ascending manifolds in pieces inside the cells formed by the descending manifolds. In other words, the structure of the descending manifolds is used while computing the ascending ones in order to maintain the structural integrity of the whole complex. Note that it is not guaranteed that the same complex would be obtained if first the ascending, and then the descending manifolds were computed. An implementation of this algorithm is described in [NP05]. The running time of the algorithm for computing the Morse-Smale complex is bounded from above by the time for sorting the vertices, plus the input size, for constructing and analyzing the vertex links, plus the output size for describing the resulting Morse-Smale complex, as pointed out in [EHNP03]. Sorting the  $n$  vertices of the input model takes  $O(n \log n)$  time, the input size is  $O(n^2)$ , while the worst case for the size of the output can be arbitrary large.

In [CCL03], an approach, rooted in the discrete Morse theory proposed by Forman [For98, For02a], is presented for the computation of the Morse-Smale complex for a two-dimensional simplicial complex. In order to apply Forman theory to a scalar field  $f$  which is given only at the vertices of a mesh,  $f$  is suitably extended by defining it on all 1- and 2-simplices (i.e. edges and triangles), in such a way that minima, saddles and maxima of  $f$  occur at vertices, edges, and triangles, respectively. A discrete gradient vector field is induced on the complex by the discrete Morse function [For98, For02b]. The algorithm proposed in [CCL03] is based on the analysis of the 1-skeleton of the simplicial complex (i.e. the graph formed by its vertices and edges) and the dual graph of the underlying triangle mesh (in which the nodes correspond to the triangles and the arcs correspond to the edges). The ascending manifold of a minimum  $p$  consists of vertices and edges, while the descending manifold of a maximum  $q$  is made of triangles and edges. In [LLT03, LLT04], it has been shown that a gradient vector field is equivalent to two spanning forests, one on the 1-skeleton and the other on the dual graph of the model, such that an edge cannot belong to both forests. The roots of the two forests are the minima and the maxima, respectively. The connected components of the two forests define the descending and ascending 2-manifolds. The Morse-Smale complex is obtained as the intersections of these regions. This algorithm can be classified both as a boundary-based and as a region-based technique, since the ascending 2-manifolds are computed through a sort of region-growing approach around the maxima, while the boundaries of the descending 2-manifolds are computed from the forests in the 1-skeleton. The worst-case time complexity of the preprocessing edge sorting step is  $O(n \log n)$ , and that of the forest creation step is  $O(n\alpha(n))$ , where  $n$  denotes the number of vertices of the model, and  $\alpha$  is the inverse of the Ackermann function [Ack28, CCL03].

Algorithms exist in the literature that compute the Morse-Smale complex directly from two-dimensional regular grids [BPS98, SW04, Sch05]. They are boundary-based in nature, since they compute the 1-skeletons of the Morse-Smale complex (i.e. the critical net), through a

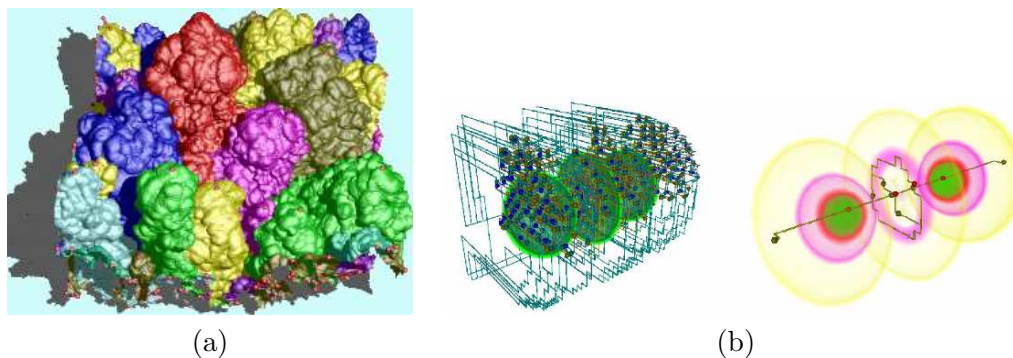
Morse and Morse-Smale complexes			
Algorithm	Input	Output	Costs
[VS91]	Regular	Morse complex	$O(n \log n)$
[Mey94]	Regular	Morse complex	$O(n \log n)$
[TIS <sup>+</sup> 95]	Simplicial (2D)	Morse-Smale complex	$O(n) + O(n \cdot c_p)$
[BS98]	Simplicial (2D)	Morse-Smale complex	$O(n \log n)$
[BPS98]	Regular (2D)	Morse-Smale complex	$O(n)$
[BPS98]	Regular (3D)	1-skeleton of the Morse-Smale complex	$O(n \log n)$
[MW99]	Simplicial (2D)	Morse complexes	$O(n \log n)$
[SS00]	Simplicial (2D/3D)	Morse complexes	$O(n \log n)$
[EHZ01]	Simplicial (2D)	Morse-Smale complex	$O(n \log n)$
[DGG03]	Simplicial (3D)	Morse complexes (stable manifolds)	$O(n \log n)$
[DDM <sup>+</sup> 03a]	Simplicial (2D)	Morse complexes	$O(n)$
[DDM03b]	Simplicial (2D/3D)	Morse complexes	$O(n)$
[BEHP03]	Simplicial (2D)	Morse-Smale complex	$O(n \log n)$
[WSH03]	Regular (3D)	Critical regions for maxima and minima	$O(n)$
[CCL03]	Simplicial (2D)	Morse-Smale complex	$O(n \log n + n\alpha(n))$
[EHNP03]	Simplicial (3D)	Morse-Smale complex	$O(n \log n + n\alpha(n))$
[NGH04]	Simplicial (2D)	Morse complexes	$O(n \log n)$
[Pas04]	Simplicial (2D)	Morse-Smale complex	$O(n \log n)$
[SW04]	Regular (2D)	Morse-Smale complex	$O(n)$
[Sch05]	Regular (2D)	Morse-Smale complex	$O(n)$
[MDD <sup>+</sup> 07]	Simplicial (2D)	Morse complex	$O(n)$

**Table 2.1:** Algorithms for the extraction of the descending/ascending Morse complex or of the Morse-Smale complex. For each algorithm the input model is outlined (regular or simplicial) as well as the output it produces (Morse or Morse-Smale complex).

technique conceptually very similar to the one used for two-dimensional simplicial models. As in the case of simplicial models, the problem of computing a Morse-Smale complex from 3D regular models has not been studied extensively. There are few algorithms that extract critical points [BPS98, WSHH02, WSH03, WS04].

### 2.1.3 Computational complexity

Table 2.1 summarizes the computational complexity of the main algorithms for Morse and Morse-Smale complex computation. The running time complexity is not specified by several of the authors and is therefore not reported in the table. However, the time complexity varies from  $O(n)$  to  $O(n \log n)$ , where  $n$  is the number of vertices in the initial model. An  $O(n)$  implementation is possible for several algorithms which do not require an initial sorting step.



**Figure 2.5:** (a) Visualization of the bubble structures in a Rayleigh-Taylor instability problem [LBM<sup>+</sup>06]. (b) The initial Morse-Smale complex of spatial probability distribution of electrons in a hydrogen atom under a large magnetic field and the Morse-Smale complex features after simplification.[BPH05]

### 2.1.4 Applications

Applications of Morse and Morse-Smale complexes can be found in scientific visualization [BEHP03, Pas04, EHZ01, EHNP03, GNP<sup>+</sup>05, GNP<sup>+</sup>06, LBM<sup>+</sup>06] (Figure 2.5), where scientific data consist of measurements over a geometric domain or space. Applications in physics simulation of the turbulent mixing between two fluids are discussed in [BP07]. The segmentation provided by these complexes allows modeling the *bubbles* formed during the mixing process (Figure 2.5(a)). Some methods have been applied for segmenting and analyzing molecular 3D shapes [CCL03, NWB<sup>+</sup>06, BP07] to study the role of cavities and protrusions in protein-protein interactions.

The computation of ascending manifolds of the distance function has been used also in shape matching and retrieval [DGG03]. Since the method is tuned on the use of the distance function, the resulting segmentation extracts the protrusions of the shape. Based on such segmentation, a shape signature is defined which associates with each Morse complex of the segmented 2-manifold a number of properties, e.g. the weighted volume or the bounding box, that are used for similarity assessment. The results are mainly geared towards shapes that exhibit a structure that is well-described by protrusions.

Interesting results have also been reported for the analysis of terrains in Geographic Information System applications [TIS<sup>+</sup>95, DDM<sup>+</sup>03a, BEHP03]. For example, in [DDM<sup>+</sup>03a], the extraction of the Morse-Smale complex was applied to generate a multi-resolution model for terrains which encodes the terrain morphology at a continuous range of different resolutions.

In the field of image analysis, watershed algorithms have been used for image segmentation [VS91, Mey94, NS96, BM00]. Watershed approaches have been applied to 3D shape segmentation, as for example in [Pag03, MW99], where the curvature is used as the height function in order to obtain *natural* shape segmentations from a human perception point of view. The

algorithm in [MDD<sup>+</sup>07] has been applied to curvature-based shape segmentation, based on the discrete curvature estimation technique in [MDDP07]. These approaches have important applications in form feature extraction, mesh reduction, and texture mapping. Scalar topology detection has been proven to be useful for image co-registration, iso-contouring, and mesh compression [BS98, BPS98].

Recent research activities have moved towards *hierarchical representations* of scalar fields. For example, the Morse-Smale complex has been used to perform controlled simplification of topological features in functions defined on two-dimensional domains [BEHP04, EHZ03, DFPV06]. These works are motivated by two major issues.

The first issue is a common problem in both image and mesh segmentation algorithms and is the over-segmentation due to the presence of noise in the data sets. For this purpose, *generalization algorithms* have been developed by several authors that locally simplify the structure of a Morse-Smale complex [Wol04, EHZ01, BEHP04, TIS<sup>+</sup>95, Tak04, GNP<sup>+</sup>05, GNP<sup>+</sup>06, NWB<sup>+</sup>06]. In [LBM<sup>+</sup>06], for example, the authors build a hierarchical Morse-Smale complex for the envelope surfaces describing the boundary between undisturbed and mixed fluids, and they use topological persistence (see Section 2.3.2) to automatically cleanup the noise.

The generalization of a Morse-Smale complex for a 2D scalar field consists of collapsing a maximum-saddle pair into a maximum, or a minimum-saddle pair into a minimum, so as to maintain the consistency of the underlying complex. This operation is called *cancellation*. A generalization can be described in terms of the combinatorial representation of the critical net, defined by the surface network [DFPV06]. The problem of generalizing 3D Morse-Smale complexes has been recently investigated in [EHNP03, GNP<sup>+</sup>05, GNP<sup>+</sup>06], by defining three cancellation operators and extending the 2D technique described in [EHZ01, BEHP04].

The second issue is related to the large size and complexity of available scientific data sets. Thus, a multi-resolution representation is crucial for an interactive exploration of such data sets. There exist just a few proposals in the literature for multi-resolution representations for 2D scalar fields based on Morse-Smale complexes [BEHP04, BPH05, DFPV06, DDV07]. All such proposals are based on the general multi-resolution framework for cell complexes introduced in [DMP99].

Starting from the eigenfunctions of the Laplacian matrix of a closed triangle mesh, the Morse-Smale complexes are used as quadrangular complexes and applied to mesh parameterization and remeshing in [DBG<sup>+</sup>06]. The Morse-Smale complexes are simplified through cancellation operations of the critical points until a *denoised* complex is obtained. Then, the simplified base complex is used to build a global parameterization over the complex. Once possibly degeneracies of the complex were eliminated, the quadrangular complex is used to produce a semi-regular mesh.



## 2.2 Graph-based shape descriptors

In his pioneering work, Cayley [Cay59] highlighted the relevance of contour lines for topographic analysis. Cayley defined contour lines to be connected sets of points at which the elevation field has some specific constant value. Similar concepts were extended by Maxwell [Max70]:

The results of the survey of the surface of a country are most conveniently exhibited by means of a map on which are traced contour-lines, each contour-line representing the intersection of a level surface with the surface of the earth, and being distinguished by a numeral which indicates the level surface to which it belongs.

Contour trees have been used mainly to study the shape of scalar fields, and no distinction is made between the shape and the function used to analyze it: both coincide with the scalar field itself. Contour trees describe the shape of a scalar field  $f$  by analyzing the evolution of its level sets, as  $f$  spans the range of its possible values: components of level sets may appear, disappear, join, split, touch the boundary or change genus. The contour tree stores this evolution and provides a compact description of the properties and structure of the scalar field. Contour trees, however, could, in principle, be defined for any shape with any mapping function, and the theory behind them is general. Contour trees, in all their variants, are discussed with emphasis on the methods developed in Computer Graphics and with pointers to similar structures defined in Computer Vision.

The generalization of a contour tree is given by Reeb graphs, even if their definition is slightly different, as presented in the literature. While the definition and use of contour trees developed mainly as an answer to computational issues, Reeb graphs have a more theoretical nature. Their definition and theoretical study date back to 1946, thanks to the research work of a French mathematician, George Reeb. With respect to the modularity of Morse theory, Reeb graphs are the first example of a fully modular framework for studying the shape of a manifold: here the shape exists by itself and the function used to study it can be arbitrarily chosen. In recent years, Reeb graphs have become popular in Computer Graphics as a tool for studying shapes through the evolution and arrangement of the level sets of a real function defined over the shape. Reeb graphs effectively code the shape, both from a topological and geometrical perspective. While the topology is described by the connectivity of the graph, the geometry can be coded in a variety of different ways, according to the type of applications the Reeb graph is devised for.

Contour trees and Reeb graphs are frequently associated, in the literature, with the concept of skeletal graphs or centerline skeletons. By coding the centroid of each iso-contour, it is indeed very easy to trace a kind of centerline spanning the volume enclosed by the shape. Centerline skeletons are very popular in Computer Graphics and Vision, and are, in principle,

related to the medial axis transformation, in the sense that they represent an effective way of reducing a complex 3D shape to a simple one-dimensional geometric abstraction [CSM05, LLS92].

### 2.2.1 Contour tree

Starting from the consideration that the evolution of contour lines on a surface explicitly represents hills and dales with their elevation-based adjacency relationships, contour trees were originally introduced as an efficient data structure to store containment relationships among contours in contour maps, typically representing terrain elevations or any other continuous real function of two variables [BR63].

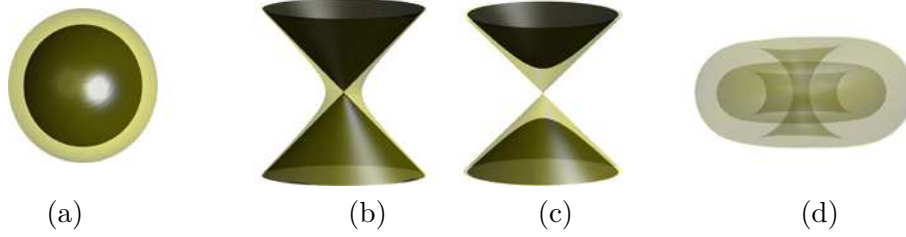
The main target of contour trees was the fast evaluation of elevations at locations other than the points on the contours [FM67, Mer73]. In the last decade, contour trees have been extensively used as a more general tool to analyze and understand shapes defined by  $n$ -dimensional scalar fields, as a support to scientific visualization of complex phenomena. In their general form, contour trees explore the shape of a scalar field  $\Gamma = (D, f)$  by analyzing the evolution of its level sets as  $f$  spans the range of its possible values over  $D$ : isocontours may appear, disappear, join, split, touch the boundary or change genus. The contour tree stores this evolution and provides a compact description of the properties and structure of the scalar field.

#### 2.2.1.1 Theoretical aspects

In the literature, several slightly different definitions of contour trees have been introduced. All of them reflect the intuition that each connected component of the level sets of the scalar field is contracted to a point and the contour tree represents the *events* in their evolution, as the isovalue varies in the range of possible values. These events, which correspond for example to the creation, union, or disappearance of isocontours, are closely related to the presence of critical points of the scalar field. In a more general case it has been demonstrated that, given a smooth 2-manifold  $M$  and a Morse function  $f$  on  $M$ , when the isovalue spans a range of values containing a critical value of  $f$  then the isocontours change (see [Gra71]). In contrast, a change in the topology of the level sets locates a critical value of  $f$ . The same results can be derived for piecewise linear functions [CLLR05].

The differences in the surveyed definitions of contour trees mainly depend on the type of evolution, that is on the type of critical point, stored in the structure. The contour tree typically keeps track of the critical points in which *only* the number of components of the level set varies, but not the genus of isocontours. For two-dimensional scalar fields this situation does not occur, but for three-dimensional scalar fields there are critical values at which the topological genus of the isosurface changes without modifying the number of

connected components nor the adjacency of the isocontours, see Figure 2.6.



**Figure 2.6:** Isosurfaces around a minimum (a) and a saddle (b-c). At the saddle point in (d), a torus evolves into a sphere changing the genus of the isosurface without altering the number of components (1) of the level set.

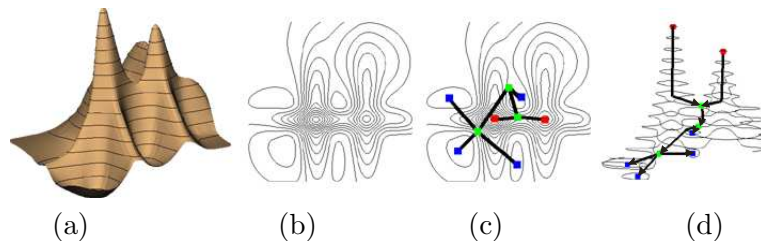
In this Chapter, we have decided to adopt the term *component-critical points* to denote critical points at which only the number of connected components of the level set varies, as used for example in [CLLR05]. In the literature, the terminology used to distinguish different types of critical points with respect to changes in the level sets is quite inhomogeneous: for example, in [Car04] critical points are named *Morse-critical* points while the component-critical points are simply called critical points.

In the following, we use a Morse-theoretic definition of contour tree inspired by the one proposed in [Car04]. Given a scalar field  $\Gamma = (D, f)$ , with  $f$  Morse, two isocontours  $\mathcal{C}$  and  $\mathcal{C}'$  are said to be *equivalent* if there exists some  $f$ -monotone path  $\alpha$  in  $D$  that connects some point in  $\mathcal{C}$  with another in  $\mathcal{C}'$  such that no point  $x \in \alpha$  belongs to a contour of any component-critical points of  $f$  [Car04]. The classes induced by this equivalence are called *contour classes*. Contours that include critical points are the sole members of their (finite) contour classes. In contrast, infinite contour classes correspond to open intervals and represent a set of contours of essentially identical connectivity. Then, the *contour tree* is a graph  $(V, E)$  such that:

1.  $V = \{v_i | v_i \text{ is a component-critical point of } f\}$ ;
2. for each infinite contour class created at a component-critical point  $v_i$  and destroyed in another component-critical point  $v_j$ ,  $(v_i, v_j) \in E$ .

Finally, it is assumed that an arc  $(v_i, v_j)$  is directed from the higher to the lower value of  $f$  on it. Figure 2.7 shows the contour tree of a two-dimensional scalar field.

It should be noted that, while the domain  $D$  of the scalar function  $f$  is supposed to be any compact subset of  $\mathbb{R}^n$ , in practice, only simply connected domains are considered, like rectangles or parallelepipeds [PCM03, CLLR05]. This directly influences the type of connectivity that the contour tree may assume, since cycles cannot appear and therefore the structure is that of a tree.



**Figure 2.7:** A two-dimensional scalar field (a,b) and its contour tree (c,d). The edge orientation and the spatial embedding of the contour tree are shown in (d).

The contour tree may be also viewed as the dual graph of the regions bounded by the level sets and by the boundary of  $D$  [Car04]. Contours that intersect the boundary of the domain  $D$  may be thought of as manifolds with boundary. Two main approaches to the analysis of the boundary have been proposed: a local classification of the boundary vertices, as in [CLLR05], or the setting of the function  $f$  at  $-\infty$  (or  $+\infty$ ) outside  $D$  [CKF03]. The latter case is equivalent to closing the boundary with a virtual root of the graph [TIS<sup>+</sup>95, BFS00], and making the image of the scalar field homeomorphic to a (hyper)sphere [Gri76]. Since critical points (and therefore the nodes of the contour tree) depend on the interpretation chosen, these two approaches generate contour trees that differ along the contours that intersect the boundary [MM05]. In particular, the virtual closure forces an interpretation of the scalar field that could be a limit when both positive and negative values of the scalar field are significant.

Several variations of the contour tree have been proposed in the literature. For example, when resolving all multiple critical points into simple ones, Takahashi et al. [TNTF04] named the contour tree of a 3D scalar field as the *volume skeleton tree*. Moreover, the contour tree may be enriched with further information on all topological changes of the level sets by adding nodes that correspond to critical values where not the number but the topology of the contours changes. This tree was first introduced in [PCM02] with the name of *augmented contour tree*, and renamed *contour topology tree* by [CLLR05] to avoid confusion with the augmented contour tree described in [CSA00], which denotes the refinement of the contour tree with the inclusion of all isocontours traced at the vertices of the input mesh. Other variations of the contour tree are the *criticality tree* [CKF03], which corresponds to the subpart of the contour tree that codes only the contour joins, and the *topographic change tree* [GHF90]. Finally, in the context of image processing, analogous structures are the *region-based contour tree* [MM05], the *component tree* [CB97, Jon99], the *fast level lines transform* [MG00], the *max tree* [SOG98] and the *scale-tree* [BHH98].

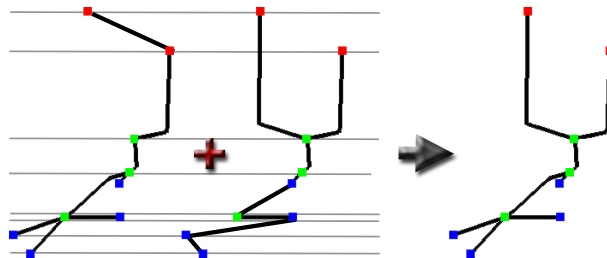
### 2.2.1.2 Computational aspects

Algorithms for the computation of contour trees first appeared in the field of spatial data handling for coding the evolution of contour lines in topographic data. One of the first papers on this topic considers the nesting relationships of a set of polygonal contours manually extracted from a topographic map [BR63]. The whole surface is enclosed by an outside region, so that each contour has an inside and outside region. Nodes representing contours are added to the tree one at time.

A more systematic approach to encode geographical data organized in a triangulation was also proposed in [dBvK97]. After ordering the function values, the authors extract contours and keep track of their evolution by sweeping the data set twice: first, from the highest to the lowest isovalue, and then sweeping again in the reverse direction from the lowest to the highest. This method is specialized for two-dimensional scalar fields and runs in  $O(n \log n)$  operations, where  $n$  is the number of vertices of the mesh. A simplification of the algorithm and its extension to higher dimensions was proposed in [vKvOB<sup>+</sup>97]. The complexity remains  $O(n \log n)$  for two-dimensional scalar fields and becomes  $O(N^2)$  for higher dimensions, where  $N$  is the number of cell higher-dimensional simplices of the mesh. In this class of methods, the vertices and the edges of the tree are called *super-nodes* and *super-arcs*, while nodes are introduced along super-arcs to represent regular points. The function  $f$  is supposed to be general in order to guarantee that all super-nodes are simple, i.e., the function  $f$  is injective on them. If  $f$  is not general, the original data are perturbed. The algorithm in [vKvOB<sup>+</sup>97] has been further improved for 3D scalar fields by Tarasov and Vyalıi [TV98] by showing that the re-labeling process can also be done efficiently in three dimensions (it requires  $O(N \log N)$  operations), and by extending the pre-processing of complex saddles.

Carr et al. [CSA00, CSA03, Car04, CS07] propose a generalization of the contour tree extraction for higher dimensions, which simplifies and extends the method in [TV98]. In this case, contours are not explicitly maintained and the pre-processing of the multi-saddles and of the surface boundary is not required. In this approach, the two sweeps are used explicitly to code the connectivity of the upper and lower level sets,  $\{x : f(x) \geq h\}$  and  $\{x : f(x) \leq h\}$ , in a join tree and a split tree, which represent the connectivity of the two sets respectively, as shown in Figure 2.8. Finally, the contour tree is assembled by picking local extrema from the join and split trees and transferring them into a so-called augmented contour tree that contains all mesh vertices. As an optional step, regular points may be removed so that the contour tree is explicitly coded. The most efficient implementation of this method requires  $O(C \log C + N\alpha(N))$  operations [Car04], where  $C$  is the number of nodes of the tree,  $N$  is the number of higher-dimensional simplices and  $\alpha$  is the inverse of the Ackermann function [Ack28].

In [PCMS04], the method in [CSA00, CSA03] is modified in order to store the contour tree in a multi-resolution fashion. The contour tree is hierarchically decomposed into a set of branches that may identify more nodes and arcs. The decomposition is not unique but, once



**Figure 2.8:** The join tree and the split tree (left) of the contour tree (right) of the scalar field in Figure 2.7.

a hierarchy has been extracted, it is possible to generate different approximations of the original tree by adding branches to a root. To obtain this representation, the merge of the join and split trees is modified considering as atoms of the operations the branches of the trees instead of the leaves. In particular, branches to be inserted in the hierarchical contour tree are selected according to a priority queue that measures the length (i.e. the difference in function values of its end-points) of each branch. Even if the algorithm requires a multi-resolution data structure, the procedure for computing the hierarchical contour tree has the same complexity of [CSA00], that is  $O(N \log N)$ . However, this cost may be improved to  $O(N)$  by using a FIFO queue instead of a priority queue, eventually generating an unbalanced tree, as discussed in [PCMS04].

An algorithm that computes the contour tree for 2D and 3D scalar fields optimally, both in space and time, has been proposed in [CLLR05]. Here the sweep algorithm described in [CSA03] is slightly modified, combined with an analytic approach and changed in the way the join and the split tree are obtained. Instead of ordering all mesh vertices, Chiang et al. first characterize the critical points, and then sort and connect them through monotone paths. The component-critical points are identified through an initial unordered scan of the vertices that analyses the neighborhood of each vertex. Only the component-critical points are ordered, and this improves the computational complexity to  $O(N + c \log c)$ , where  $N$  is the number of higher-dimensional simplices and  $c$  the number of component-critical points of the mesh. Therefore the time complexity depends on the output size. In particular, [BS04] proposed an encoding technique that, beside the contour tree of the whole domain, is able to return also a representation of the contour tree of its sub-domains.

Another extension of the algorithm in [vKvOB<sup>+</sup>97] to 3D domains has been proposed in [PCM02] and further extended in [PCM03]. In this case, the domain  $D$  is split into subparts and the isosurfaces are separately processed using a parallel approach, based on a divide-and-conquer paradigm. This paradigm is used to compute both the join and the split trees (it is worth noticing that in these papers the notions of join and split tree are used inversely from [CSA03]). In addition, a more detailed characterization of the contours is achieved coding the Betti numbers associated to each arc of the tree. Finally, the different trees are merged according to adjacency of the subparts of  $D$ , eventually revising the Betti numbers of

each component. The resulting tree is called *augmented contour tree* because all the critical points of the scalar field correspond to nodes of the tree and the contours associated with an arc contain no critical points. The complexity of the algorithm for a 3D regular mesh, that is a triangle mesh obtained from a regular grid, is  $O(n + c \log n)$  where  $n$  is number of vertices of the mesh and  $c$  is the number of critical points. Even if the method works on meshes whose vertices do not lie on a grid, the split of the mesh into subparts for the parallel computation does not improve the complexity of this method, which still is  $O(n \log n)$ , see discussion in [CLLR05].

The method proposed in [TIS<sup>+</sup>95, Tak04] is devised to build contour trees of two-dimensional scalar fields represented by a regular grid, that is converted to a triangulation by splitting each cell with its diagonal. Note that the authors refer to their contour structure as a Reeb graph. A surface network, see Section 2.1.2.1, is extracted from the grid and used as intermediate structure for constructing the contour tree. In fact, the authors prove that the contour tree of a two-dimensional scalar field may be deduced by its surface network. This method is mainly analytic because it identifies the critical points on the vertices of the triangulation by analyzing the star of each vertex and simulates the paths of steepest descent on the model. A global virtual minimum acts as the root of the tree giving a unique interpretation of the surface behavior along its boundary. The computational complexity of this algorithm was not stated by the authors, but its analysis is given in [CLLR05], where it is claimed that the method requires  $O(N)$  operations for reading the data,  $O(nc)$  operation for finding the paths and  $O(c^2)$  time for constructing the tree, where  $N$ ,  $n$  and  $c$  are respectively the number of triangles, vertices and critical points of the mesh. This approach has been extended to volumetric gridded data [TTF04] represented by tetrahedral cells, using a voxel flow network as support during the contour tree extraction instead of a surface network. Moreover, the method admitted the removal of critical points whose relevance (in the sense of length of the adjacent arcs) is irrelevant to describe the global topological structure of the volume. Even if the authors do not provide the computational complexity of their algorithm, it seems reasonable that the computational cost may be expressed with the same expression as in the two-dimensional case, considering that the number  $N$  of triangles is related also to tetrahedra. Finally, the method in [TNTF04] combines three approaches: the extraction of the contour tree in [CSA03] is joined together with the analysis of the genus of each isosurface component proposed in [PCM02] and simplified according to the procedure described in [TTF04].

The method proposed in [IK95] automatically detects a graph from a three-dimensional mesh. The authors remove simplices (i.e. vertices, edges, faces and tetrahedra) while preserving the connection between the mesh critical points. In this case the nodes of the tree might differ from the critical points and the edges are compound of a generic monotonic path between two nodes. However, this method is computationally efficient because it is linear and performs in  $O(n)$  operations, where  $n$  is the number of mesh vertices.

The approach proposed in [BFS00] extracts a contour tree, called an *Extended Reeb Graph*

(*ERG*), from two-dimensional scalar fields described by a set of isocontours. Starting from these contours, a Delaunay constrained triangulation is built, forcing the isocontours (in the sense of both edges and vertices) to belong to the mesh. The function  $f$  of the scalar field (in this case the height of the vertices) is not required to be either Morse or simple and, therefore, the method can deal with multi-saddles, plateaus and volcano rims. Similarly to [TIS<sup>+</sup>95], the surface is virtually closed introducing a global virtual minimum. Since it is supposed that all vertices in the mesh lie on contours, the authors use a characterization of the triangles which is based on the number of vertices with the same elevation. Based on this characterization, *critical areas* are defined as areas containing critical points. Then, the tree is built using a region growing process that starts from the critical areas, spreads them in all directions and completely covers the surface keeping track of the contours crossed. Unlike the approach in [vKvOB<sup>+</sup>97], the region growing process is based on a visit of mesh triangles which is linear in the number of triangles,  $O(N)$ . Therefore, the computational complexity of the method depends on the construction of the Delaunay triangulation constrained to contours, that requires  $O(n \log n)$  operations, where  $n$  is the number of vertices of the isocontours in input. An extension of this approach has been proposed in [BFS04]: here the classification of the critical areas and the graph construction do not require that the triangle mesh is originally constrained to the level sets. However, the successive insertion of the contours in the mesh increases the computational complexity of the method to  $O(\max(m + n, n \log n))$ , where  $n$  is the number of vertices of the original mesh and  $m$  is the number of vertices added during the insertion of the level sets (in the worst case  $m$  may be  $O(n^2)$ ). The extension of the method to closed surfaces [ABS03] and to generic two-dimensional surfaces [Bia04b] is discussed in Section 2.2.2.2, which is devoted to Reeb graphs.

Cox et al. [CKF03] propose a variant of the contour tree called the *criticality tree* which is based on the analysis of the isosurfaces without relying on the classical Morse theory. In particular, the authors develop a discrete theory, called *digital Morse theory*, in order to disambiguate the characterization of the isosurface evolution on cubic grids and to include non-general functions. In practice, the criticality tree proposed in this work is a join tree augmented by the component-critical points rather than a contour tree. In fact the criticality tree stores the evolution of the volumes that, starting from maxima, are bounded by the isosurfaces. These volumes are denoted topological zones and are locally nested. Due to the need of extracting the topological zones, the computational complexity of this method is  $O(kN \log(kN))$ , where  $N$  is the number of 3-cells and  $k$  is the length of the longest path traversed in the tree.

In the field of image processing, several variations of the contour tree have been adopted. The *region-based* contour tree proposed in [MM05] transposes the classical contour tree definition to gray-level images. The key idea behind this method is that the changes in the isosurfaces (i.e. the variations in the number of connected components of pixels at a given gray-level) are directly encoded in a tree. Therefore, the nodes of the tree correspond to



regions of pixels. The algorithm adopts an extraction procedure analogous to that proposed in [CSA03] which runs in  $O(p \log p)$  operations, where  $p$  is the number of pixels of the image. However, since the pixels assume only integer values, the authors claim that, if the gray-level values vary between 8 and 16, a bucket-sort algorithm to order the pixels would decrease the computational cost to be  $O(p)$ .

The *component tree*, also known as *confinement tree* or *dendrone* [Jon99, CB97, MD00], displays all image components in a hierarchical sequence. Gray-levels are therefore piled one on the other in a graph, which is a useful structure for further filtering. This hierarchy implies that the component tree is a variation of the join tree rather than a contour tree. This description has been successfully used for image segmentation and has been proved to achieve better results in comparison with standard connected filters [BJ96]. An efficient algorithm for the extraction of component trees is proposed in [MD00] where the global computational complexity performs in  $O(p \log p)$  operations, where  $p$  is the number of pixels of the image. Closely to the component tree, the *Fast Level Lines Transform (FLLT)* codes the evolution of the connected components of the gray levels of an image [MG00]. Notice that the sorting of the components is obtained checking the geometric inclusion of the level sets, thus discarding the arc orientation induced by the increase/decrease of the gray-level intensity.

Another popular structure in image processing is the *max-tree* introduced by Salembier et al. [SOG98], and its dual, the *min-tree*. Similarly to component tree, the max-tree encodes in a tree the hierarchy of the connected level set of pixels. The main difference between a component tree and a max-tree does not involve the topology (i.e., the connection among the nodes), but the construction process and the kind of information stored in each node. An efficient method for storing and building the max tree has been proposed in [HFS03]. This method performs in linear time in the number of pixels  $O(p)$  and combines the dynamic allocation of the memory in a linked list with the node tree storage provided by a hash table. The *scale-tree* in [BHH98, BHH98] is also based on the level-set image decomposition. In particular, the scale-tree automatically selects a number of scales and codes the regions with slightly different attributes, like their amplitude, shape and position so that the coding is invertible.

More recently, the interest for time-dependent data sets has increased leading to the introduction of methods for extracting the graph for time-varying models [Szy05, SB06]. The main innovation of these methods is not in the extraction algorithm itself (they refer to [PCM03] and [CSA03], respectively) but in the way contours of 2D and 3D data sets join and split during a time interval. These methods also investigate the relationships between the contour tree over the entire domain  $D$  and those restricted to its sub-domains. Contour trees are computed in a pre-processing phase and the focus is the definition of efficient structures and algorithms for the query execution. For example, the computational complexity of the query algorithm in [Szy05] is  $O(s(1 + \log(t_1 - t_0)))$ , where  $s$  denotes the maximum size of a contour tree and  $t_0, t_1$  are two time values. In [SB06] the correspondence between the

---

nodes of the contour tree is stored in a graph called a *topology change graph* that supports visualization of contour evolution. The computational complexity for extracting the topology change graph is  $O(n \log n + N + (c_t)^2 c_{t+1})$ , where  $n$  and  $N$  are respectively the number of vertices and tetrahedra of the mesh and  $c_t$  is the number of critical points of  $f$  at time  $t$ .

### 2.2.2 Reeb graph

Similar to contour trees, the main idea behind Reeb graphs is to encode the evolution and the arrangement of the level sets of a real function defined on a shape. While the definition and use of contour trees developed mainly as an answer to computational issues, Reeb graphs have a more theoretical nature. They originated in 1946 in the work of a French mathematician, George Reeb. In recent years, Reeb graphs have become popular in Computer Graphics as tools for shape description, analysis and comparison.

Reeb graphs present a framework for studying the shape of a manifold: here the shape exists by itself and the function used to study its shape can be arbitrarily chosen. Different functions can be used for extracting a structure that effectively codes the shape from both a topological and geometrical perspective. Topology here means that the shape can be described as a configuration of parts that are *attached* together respecting the topology of the shape, while geometry means that the different parts correspond to features of the shape, as embedded into the Euclidean space, that have specific properties and descriptive power (e.g., protrusions, elongated parts, wells).

#### 2.2.2.1 Theoretical aspects

Reeb graphs were first defined by Georges Reeb in 1946 [Ree46] as topological constructs. Given a manifold  $M$  and a real-valued function  $f$  defined on  $M$ , the simplicial complex defined by Reeb, conventionally called the Reeb graph, is the quotient space defined by the equivalence relation that identifies the points belonging to the same connected component of level sets of  $f$ . Under some hypotheses on  $M$  and  $f$ , Reeb stated the following theorem, which actually defines the Reeb graph.

**Theorem 2.2.1** *Let  $M$  be a compact  $n$ -dimensional manifold and  $f$  a simple<sup>2</sup> Morse function defined on  $M$ , and let us define the equivalence relation “ $\sim$ ” as  $(P, f(P)) \sim (Q, f(Q))$  iff  $f(P) = f(Q)$  and  $P, Q$  are in the same connected component of  $f^{-1}(f(P))$ .*

*The quotient space on  $M \times \mathbb{R}$  induced by “ $\sim$ ” is a finite and connected simplicial complex  $K$  of dimension 1, such that the counter-image of each vertex  $\Delta_i^0$  of  $K$  is a singular connected component of the level sets of  $f$ , and the counter-image of the interior of each simplex  $\Delta_j^1$*

---

<sup>2</sup>A function is called simple if its critical points have different values

is homeomorphic to the topological product of one connected component of the level sets by  $\mathbb{R}$  [Ree46, ER44].

Reeb also demonstrated the following theorems, which clarify the relations between the degree, or order, of the vertices of the simplicial complex  $K$  associated with the quotient space and the index of the corresponding critical point.

**Theorem 2.2.2** *The degree of a vertex  $\Delta_i^0$  of index 0 (or  $n$ ) is 1 and the index of a vertex  $\Delta_i^0$  of degree 1 is 0 or  $n$ .*

**Theorem 2.2.3** *If  $n \geq 3$  the degree of vertices  $\Delta_i^0$  of index 1 (or  $n - 1$ ) is 2 or 3. If  $n = 2$  the degree of vertices  $\Delta_i^0$  of index 1 is 2, 3, or 4. The degree of vertices  $\Delta_i^0$  of index different from 0, 1,  $n - 1$ , or  $n$  is 2.*

In other words, the first theorem states that leaf nodes of  $K$  can be either maxima or minima of  $f$ , while from the second theorem we can deduce that, for 2-manifolds that can be embedded in  $\mathbb{R}^3$ , the degree of vertices representing saddles is always 3.

To the extent of our knowledge, Reeb graphs were first introduced in Computer Graphics by Shinagawa et al. [SKK91] and the term Reeb graph is used to identify the simplicial complex associated with the quotient space. As a consequence of their ability to extract high-level features from shapes, since their introduction in Computer Graphics Reeb graphs have been gaining popularity as an effective tool for shape analysis and description tasks, especially in case of 2-manifolds.

The Reeb graph has been also used for the analysis of 3-manifolds with boundary. In this case the structure of the 3-manifold is studied either by introducing a virtual closure of the manifold [EHMP04], or by associating a Reeb graph to each 2-manifold boundary component of the 3-manifold and keeping track with a supplementary graph of the changes between interior and void [SKK91, SL01].

For orientable, closed 2-manifolds, the number of cycles in the Reeb graph corresponds to the genus of the manifold, and this result has been generalized in [CMEH<sup>+</sup>03], where the authors demonstrate that the number  $\beta_1(M)$  of non-homologous loops of the surface is an upper bound of the number of loops  $\beta_1(K)$  of the Reeb graph. The equality holds in case of orientable surfaces without boundary [CMEH<sup>+</sup>03], while, in general, the following relation holds:

$$g \leq \beta_1(K) \leq 2g + b_M - 1,$$

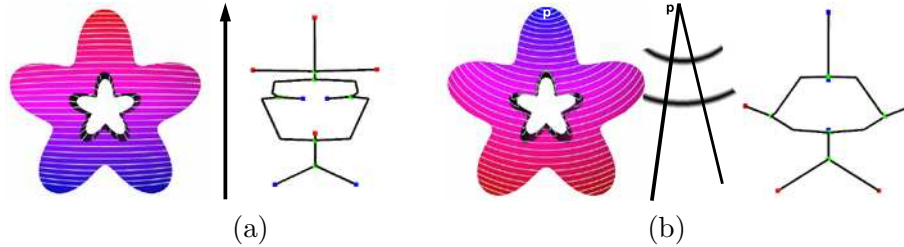
where  $b_M$  denotes the number of boundary components of the 2-manifold  $M$  having genus  $g$ . Theoretical results are available for non-orientable 2-manifolds. In this case, the number of loops of the Reeb graph verify the following relations:  $0 \leq \beta_1(K) \leq \frac{g}{2}$  when  $M$  is closed, and  $0 \leq \beta_1(K) \leq g + b_M - 1$  for manifolds with boundary.

---

As for 3-manifolds, it is not true that the number of the loops of an orientable, closed 3-manifold is independent of the mapping function  $f$ . In addition, it has been proven that for every 3-manifold  $M$  there exists at least one Morse function  $f$  such that the Reeb graph of  $M$  with respect to  $f$  is a tree [CMEH<sup>+</sup>03].

The extension of Reeb graphs to shapes defined by piecewise linear approximations has been studied by several authors. For example, in [Bia04a] the definition of Reeb graph was extended to triangle meshes representing 2-manifolds embedded in  $\mathbb{R}^3$ , with or without boundary, and which admit degenerate critical points. The relationship between the genus of the mesh and the cycles in the extended Reeb graph is maintained, as discussed in [Bia04a, Bia04b]. On the basis of this representation, a further extension of the domain of the Reeb graph to point clouds was proposed in [WXS06], defining a so-called discrete Reeb graph.

Figure 2.9 shows two examples of Reeb graphs of a closed surface. In Figure 2.9(a) some level sets of the height function are drawn with the corresponding Reeb graph; in Figure 2.9(b) the Reeb graph of the same object is shown using the Euclidean distance from a point.



**Figure 2.9:** Reeb graph with respect to the height function (a) and to the distance from a point (b).

### 2.2.2.2 Computational aspects

Several algorithms have been proposed for the computation of the Reeb graph of closed surfaces, while only a few algorithms deal with 3-manifolds, higher-dimensional or time-dependent data.

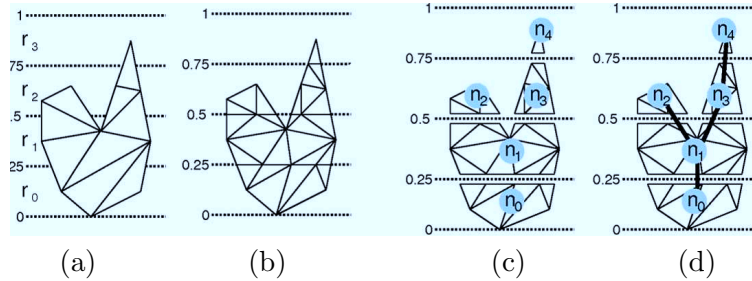
The first algorithm, proposed by Shinagawa et al. [SK91], automatically constructs the graph from surface contours generated by the height function. Since the contour ordering proposed in [BR63] is not suitable for manifolds without boundary, a weight function, which depends on the average distance between the vertices of two different contours, is defined for each pair of contours lying on adjacent (consecutive) level sets. First, the algorithm automatically generates most of the arcs of the Reeb graph where the number of contours of two consecutive cross sections is one. Then the rest of the graph is determined by using the weight function and *a priori* knowledge of the surface genus. Specifically, the graph is completed by adding edges in decreasing order of the weight between contour pairs, so that the genus of the graph preserves that of the original surface. The main drawbacks of this

algorithm are the need for *a priori* knowledge of the genus of the surface and the fact that this procedure is limited to contour levels of the height function. In addition, since this algorithm loses the shape information between two consecutive cross sections, the frequency of the contours of the surface is critical; therefore, a reasonable computation of the graph requires a high number of surface slices and it is time and space consuming ( $O(n^2)$ , where  $n$  represents the total number of vertices of the scattered contours). To deal with models with cavities, Shinagawa et al. encoded the Reeb graphs of both the shape and its complement. Similar considerations on the evolution of the model and its complement have been proposed in [SL01] to define a method for the extraction of a topological graph for cortical volume data.

A general algorithm for the Reeb graph extraction of 2-manifolds with or without boundary represented by a simplicial complex was proposed in [CMEH<sup>+</sup>03]. This approach also works for non-orientable models, like the Klein bottle. The basic assumption here is that the mapping function is Morse and simple, so that critical points have pairwise different function values. Then, the Reeb graph is constructed by storing the level sets while sweeping the domain of the function. To identify the critical points, the star of each vertex is classified according to the approach in [EHZ03]. Once critical points have been detected, all vertices of the model are processed according to the increasing value of the function  $f$  and the evolution of level sets is tracked. Since operations are done on the edges, the complexity of the algorithm is  $O(n \log n)$ , where  $n$  is the number of edges of the complex.

Recently, Pascucci et al. [PSBM07] proposed an algorithm able to compute efficiently the Reeb graph of simplicial complexes of arbitrary dimension. This algorithm is based on the assumption that the Reeb graph of the simplicial complex is equivalent to the Reeb graph of its 2-skeleton. This assumption implies that the structure obtained from this algorithm slightly differs from the original definition given by Reeb [Ree46] because it is not able to distinguish changes in the topology of the isosurfaces. For instance, the 2-skeleton of a simplicial complex of dimension 3 is not sensitive to the inner cavities of the model. The usage of a stream approach eliminates the need of an initial ordering of the mesh vertices, allows a progressive computation of the graph able to encode very large models and provides a way to compute a run-time simplification of Reeb graph loops based on their relevance. To achieve these tasks, two structures are used: one for the input 2-skeleton and the other for the Reeb graph. These two structures are related to each other and, when a new element is inserted in the 2-skeleton, the Reeb graph is consequently updated. The basic operations to update the Reeb graph are the creation of new nodes and arcs and the merge of two paths. In particular, two paths are merged if, during the progressive visit of the 2-skeleton, a hole is filled or a loop is removed from the Reeb graph. The computational complexity of the algorithm is still  $O(n \log n)$ , where  $n$  represent the number of vertices, which is the theoretical lower bound complexity for the Reeb graph extraction. However, the usage of the stream approach makes this method efficient in practice, allowing a fast computation of the Reeb graph also on large meshes.

The method proposed in [HSKK01] provides a *multi-resolution Reeb graph (MRG)* representation of triangle meshes which is independent of the object topology. The construction of the MRG begins with the extraction of the graph at the finest resolution desired, then adjacency rules are used to complete the multi-resolution representation in a fine-to-coarse order. First of all, the domain of the mapping function is divided into a number of intervals and triangles whose image under  $f$  lies in two intervals are subdivided so that the image of every triangle belongs to only one interval. Second, the triangle sets, that is the sets of connected components of triangles whose images belong to the same interval, are calculated. A node of the graph is associated with each triangle set and arcs are detected by checking the region adjacency of triangle sets. It is worth noticing that this contouring approach induces a quantization of the interval of  $f$  that, although it may locally amend the local topological noise, does not guarantee that the number of loops of the MRG equals the number of holes of the surface. Once the function  $f$  has been evaluated on the vertices of the mesh and triangles have been split, the Reeb graph extraction requires  $O(n + m)$  operations, where  $n$  represents the number of triangles of the original mesh and  $m$  represents the number of triangles inserted during the subdivision phase. Notice that the evaluation of the function  $f$  may be a computationally expensive step. For example, the exact computation of the geodesic function proposed in [HSKK01] requires  $O(n^2 \log n)$  operations, while its approximation runs in  $O(kn \log n)$ , where  $k$  is a user-defined constant (usually greater than 150). In Figure 2.10 an example of the Reeb graph construction method proposed in [HSKK01] is shown; in this case the domain of  $f$  is subdivided in 4 intervals. The contour insertion in Figure 2.10(b) determines a set of mesh regions that correspond to the graph nodes in Figure 2.10(c), while their adjacency originates the arcs of the graph (see Figure 2.10(d)).



**Figure 2.10:** Pipeline of the multi-resolution Reeb graph extraction in [HSKK01].

The *Extended Reeb Graph (ERG)* representation proposed in [ABS03, Bia04a] is able to represent a surface with or without boundary through a finite set of contour levels of a given mapping function  $f$ . In [Bia04b], the ERG was extended to surfaces having an arbitrary number of boundary components. To obtain a minimal (in the sense of graph loops) representation of the ERG, this algorithm virtually closes all boundary components, as detailed in [Bia05]. An extension of the ERG definition to unorganized point clouds of 3D scan data that represent a human body has been proposed in [WXS06]. Since a polygonal mesh is not

available, a surface is implicitly defined by assuming that the Euclidean distance among a point  $p$  and its closest point  $q$  is smaller than a given threshold  $\epsilon$ . Point sets whose sampling is sufficiently fine are connected in a discrete sense. Therefore, level sets are defined as points that share the same value of a mapping function and are connected in the discrete sense. The resulting graph is called the *Discrete Reeb Graph (DRG)*. Once a set of level sets has been detected, the graph is progressively constructed visiting all points that are linked with respect to the threshold  $\epsilon$ . The declared complexity of the graph computation mainly depends on the extraction of the level sets, which requires  $O(nk^2)$  operations, where  $n$  is the number of points and  $k$  is the average of the points contained in the neighborhood of each point.

The approach proposed in [WHDS04] works on volume data. In this case, the data are swept with a plane that generates a sequence of *slices*, which are formed by the sets of grid elements bounded by two adjacent isosurfaces. Each connected component of a slice is called a *ribbon* while the *contours* are given by the intersection of the isosurfaces with a set of slicing planes. The graph described in this approach is called an *augmented Reeb graph* because it also encodes geometric information for each contour and each ribbon. The traversal is analyzed at discrete  $z$  intervals of the volumetric grid along the boundary of a distance function and may be done out-of-core on the dataset. Isosurfaces are analyzed one slice at time and contours are constructed by searching from an arbitrary edge in the plane until the contour is closed. Similarly, ribbons are constructed through a breadth-first traversal of the slice elements, starting from the polygons adjacent to a contour until the connected component is completely detected. Both ribbons and contours correspond to nodes of the Reeb graph while their adjacency is coded in the edges. To avoid an object handle being completely contained within a ribbon, the Euler characteristic of each isosurface component is computed and, possibly, the sweep is locally refined. In this way the topology of the volume is completely coded and, in each interval, the Reeb graph structure corresponds to the Euler characteristic of the object ribbons.

In the analogous context of 3D binary images represented by voxel models, a graph similar to the Reeb graph has been proposed in [SL01]. Since in a volume model configurations with internal cavities are not fully described by the shape boundary, the 3D image is embedded in its bounding box and the graphs are extracted for both the object and its complement. A point is associated with each section. For each direction  $x$ ,  $y$  and  $z$ , a foreground connectivity graph  $G$  is extracted. Since cycles are not admitted in the final graph (because cycles denote inner cavities and handles of the model to be removed as noise), a maximum spanning tree is computed to select the set of arcs that should be eventually removed from  $G$ . Then, graph tests are performed to simplify the graph until all cycles are removed. Although the computational cost of this algorithm was not provided by the authors, the computation of the maximum spanning tree seems to be the most time consuming operation. Therefore, the complexity of this method is  $O(v \log v)$ , where  $v$  represents the number of voxels of the cortical volume.

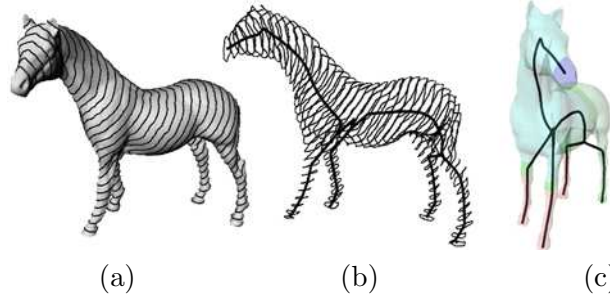
The *hyper Reeb graph* proposed in [FAT99, FTAT00] deals with 3D volume fields. The main concept behind this description is the representation of the isosurfaces of a volume in terms of their Reeb graphs, and the detection of the topology changes of these isosurfaces through the modifications of their Reeb graphs. Once the volume data set is swept in the height direction and a Reeb graph has been extracted for each isosurface, this collection of graphs is encoded in a hyper graph, whose nodes correspond to the Reeb graphs. The Reeb graph of a single isosurface is extracted from the surface network of the isosurface extending the algorithm proposed in [TIS<sup>+</sup>95] for two-dimensional scalar fields. This computation costs  $O(N) + O(nc) + O(c^2)$  operations where  $N$  and  $n$  represent the number of faces and vertices of the mesh and  $c$  is the number of its critical points (see Table 2.2 in Section 2.2.1.2). Therefore, the computational complexity of the hyper Reeb graph is quite significant and depends on the number of isosurfaces taken into account.

The spatial embedding of the Reeb graph of a surface is often regarded as a topological centerline skeleton. In this case, each contour may be visualized through its centroid. Various methods adopt a centerline encoding that can be related to Reeb graphs, even if they do not explicitly originate from the Reeb graph definition. These approaches generally do not require the mapping function to be general nor simple and, differently from the centerlines based on the distance function, act as topological centerlines. The construction of the *Level Set Diagrams (LSD)* from triangulated polyhedra proposed in [LV99] uses Euclidean distances for wave propagation from a seed point. An heuristic detects a point at the top of a protrusion on the basis of the geodesic distance. This source point automatically determines a privileged “slicing direction”. In this approach, a skeleton-like structure, available for input objects of genus zero, is proposed, which is essentially a tree made of the *average* points (in the sense of centroids) associated with the connected components of the level sets of the geodesic distance from the source. The resulting skeleton is invariant under rotation, translation and uniform scaling (see Figure 2.11). The method in [Axe99] follows the same approach replacing Euclidean distance with topological distance.

An extension of the approaches in [Axe99] and [LV99] to non-zero genus surfaces was presented in [HA03]. In this case, the evaluation of the measuring function, the mesh characterization (based on local criteria) and the construction of the graph are performed at the same time using Dijkstra’s algorithm. A similar approach was introduced in [WDSB00] for implicit storage of meshes obtained from distance volumes. In this case the graph extraction is driven by the evolution of the isosurfaces of the geodesic distance from a single point. In fact, the object topology is reconstructed by considering a wavefront-like propagation from a seed point by applying Dijkstra’s algorithm. The cost complexity of these methods, all of which require the vertices to be sorted, is  $O(n \log n)$ , where  $n$  is the number of vertices of the mesh.

Finally, the application of the approach in [LV99] to point clouds was proposed in [VL00]. The algorithm runs in  $O(e + n \log n)$ , where  $e$  is the number of edges in the neighborhood graph and  $n$  is the number of points.

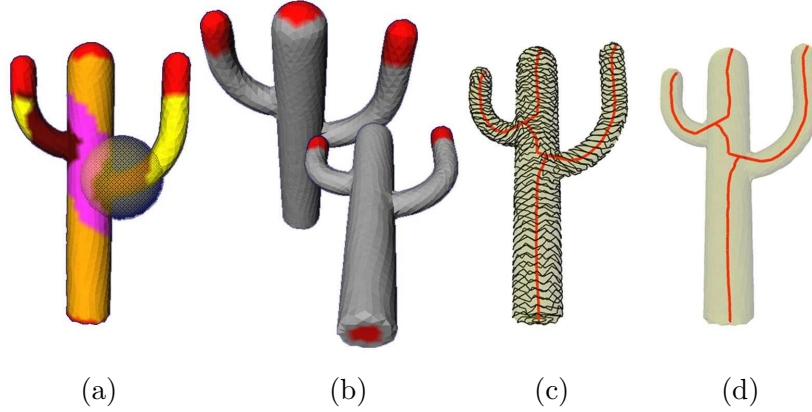




**Figure 2.11:** Level sets (a) and the centerline (b,c) of an horse using the geodesic distance from a source point as proposed in [Lazarus and Verroust 1999].

The method in [MP02] uses the multi-resolution curvature evaluation proposed in [MPS<sup>+</sup>04] to locate seed points that are subsequently used during the geodesic expansion. Seed points, also called *representative vertices*, are sequentially linked by using a wavefront traversal defined on the simplicial complex (see Figure 2.12(a,b)). Once a set of representative vertices is selected, rings made of vertices of increasing neighborhoods are computed in parallel until the whole surface is covered (see Figure 2.12(c)), in a way similar to the wave-traversal technique [AE98]. Rings growing from different seed points will collide and join where two distinct protrusions depart, thus identifying a branching zone; self-intersecting rings can appear when expanding near handles and through holes. A skeleton is drawn according to the ring expansion: *terminal nodes* are identified by the *representative vertices*, while union or split of topological rings give *branching nodes*. It is worth noticing that the terminal nodes of the graph have degree 1. Arcs are drawn joining the center of mass of all rings (see Figure 2.12(d)). Therefore, the complexity of the proposed graph, in terms of number of nodes and branches, depends on the shape of the input object and on the number of seed points which have been selected using the curvature estimation criterion. The number of operations needed for extracting the skeleton is  $O(n)$  in the number of mesh vertices (each vertex is visited once) even if an accurate evaluation of the high curvature points may require  $O(n^2)$  operations. In addition, this curve-line representation has at least as many cycles as the number of holes of the surface; however, some unforeseen cycles may appear as a result of colliding wavefronts.

Time-varying data are not always best dealt with by means of a function defined over four equivalent dimensions. In practice, most four-dimensional data consist of time-slices of three-dimensional data that can be treated with a refinement of 3D algorithms. An algorithm for studying the evolution of the Reeb graph when the mapping function varies with time is proposed in [EHMP04]. To simplify the topological complexity of the space, a point at infinity is added. In this way the space becomes topologically equivalent to the 3-sphere and each Reeb graph will be equivalent to a tree. In this framework, the time is represented by a conventional priority queue that stores birth-death and interchange events, that are prioritized by the moments in time they occur. Once a Reeb graph is computed,



**Figure 2.12:** (a) Vertex classification based on Gaussian curvature; (b) high curvature regions are depicted in red; (c) topological rings expanded from centers of high curvature regions; (d) the graph obtained as proposed in [Mortara and Patané 2002].

the evolution of the graph over time is detected using a Jacobi curve that collects the birth-death points and maintains the occurrence of a new event. A data structure is used to store the entire evolution. The computational cost of this algorithm,  $O(N + En)$ , depends on the number  $N$  of simplices of the triangulation of the space-time data, the upper number  $n$  of simplices at a time  $t$  and the amount  $E$  of birth-death and interchange events.

### 2.2.3 Computational complexity

A summary of methods for contour tree and Reeb graph extraction is proposed in Table 2.2. For every method we report the kind of output structure as named by the authors, in detail: CT is the contour tree, RG a Reeb graph, ACT an augmented contour tree, MT a max tree, ST a scale tree, CompT a component tree, CTree a criticality tree, C a centerline, MRG a multiresolution Reeb graph, AMRG an augmented multiresolution Reeb graph, ERG an extended Reeb graph and ARG an augmented Reeb graph.

<sup>1</sup>The complexity is, respectively,  $O(N \log N)$  for 2D and  $O(N^2)$  for 3D domains.

<sup>2</sup>The method is essentially the same in both papers. The cost improvement is due to a more efficient implementation of the data structures.

<sup>3</sup>The complexity of this method depends on the use of a FIFO or a priority queue.

<sup>4</sup>The complexity of this method varies if considering regular or simplicial meshes.

<sup>5</sup>The complexity of the method may decrease to  $O(p)$  if the range of pixel values is between 8 and 16.

<sup>6</sup>Once a set of seed points has been recognized, the complexity of the skeleton extraction is linear in the number of mesh vertices but an accurate evaluation of the high curvature points requires  $O(n^2)$  operations.

<sup>7</sup>The computational cost of the time varying algorithms reported in the Table refers to the complexity of the query at a given instant  $t$  rather than the complexity of the built of the structure.

Graph-based shape descriptors						
Approach	Method	Output	Domain			Costs
			2D	3D	nD	
[BR63]	Manual	CT	X			–
[IK95]	Erosion	CT	X	X		$O(n)$
[dBvK97]	Sweep	CT	X			$O(N \log N)$
[vKvOB <sup>+</sup> 97, vKvOB <sup>+</sup> 04]	Sweep <sup>1</sup>	CT	X	X		$O(N \log N) / O(N^2)$
[TV98]	Sweep	CT	X	X		$O(N + n \log n)$
[CSA00, CSA03] [Car04]	Sweep <sup>2</sup>	CT	X	X	X	$O(N \log N)$ $O(C \log C + N\alpha N)$
[PCMS04]	Sweep <sup>3</sup>	CT				$O(N \log N)$ $O(N)$
[CLLR05]	Sweep	CT	X	X		$O(N + c \log c)$
[PCM02] [PCM03]	Sweep <sup>4</sup>	ACT	X	X		$O(n + c \log n) /$ $O(n \log n)$
[TIS <sup>+</sup> 95] [Tak04]	Analytic	RG	X X			$O(N) + O(nc) + O(c^2)$
[TTF04]	Analytic	RG		X		$O(N) + O(nc) + O(c^2)$
[BFS00]	Contours	ERG	X			$O(n \log n)$
[BFS04]	Contours	ERG	X			$O(\max(m + n, n \log n))$
[CKF03]	Isosurfaces	CTree		X		$O(kn \log(kn))$
[MM05]	Level sets <sup>5</sup>	CT	2D Images			$O(p \log p)$
[MD00]	Level sets	CompT	2D Images			$O(p \log p)$
[BHHC98] [BHH98]	Level sets	ST	2D Images			$O(p \log p)$
[HFS03]	Level sets	MT	2D Images			$O(p)$
[Szy05]	Level sets	CT	Time varying data <sup>7</sup>			$O(s(1 + \log(t_1 - t_0)))$
[SB06]	Analytic	CT	Time varying data <sup>7</sup>			$O(n \log n + N + (c_t)^2 c_{t+1})$
[SK91] [SKK91]	Contours	RG	X	X		$O(n^2)$
[HSKK01] [BRS03] [TS04]	Contours	MRG MRG AMRG	X			$O((n + m))$
[ABS03] [Bia04b, Bia05]	Contours	ERG	X			$O(\max(m + n, n \log n))$
[CMEH <sup>+</sup> 03]	Analytic	RG	X			$O(n \log n)$
[PSBM07]	Analytic	RG	X	X	X	$O(n \log n)$
[Axe99] [LV99] [HA03]	Contours	C RG	X			$O(n \log n)$
[MP02] <sup>1</sup>	Contours	SG	X			$O(n)$
[WDSB00]	Contours	C		X		$O(n \log n)$
[WHDS04]	Contours	ARG		X		$O(n \log n)$
[SL01]	Contours	C		X		$O(v \log v)$
[VL00]	Contours	C	Point clouds			$O(e + n \log n)$
[WXS06]	Contours	RG	Point clouds			$O(n)$
[EHMP04]	Analytic	RG	Time varying data <sup>7</sup>			$O(N + En)$

**Table 2.2:** Classification of the methods that extract graph-based descriptors. Symbols:  $N$  is the number of higher-dimensional simplices or cells;  $n$  is the number of vertices or points;  $m$  is the number of vertices inserted in the mesh during contouring phases;  $c$  is the number of critical points;  $C$  is the number of tree nodes;  $\alpha$  is the inverse of the Ackermann function;  $k$  is the length of the longest path traversed in the tree,  $p$  is the number of pixels,  $v$  is the number of voxels  $t$  is time,  $e$  is the number of edges in the neighborhood tree and  $E$  is the amount of birth-death and interchange events. Note that, in the 2D case,  $N = O(n)$ .

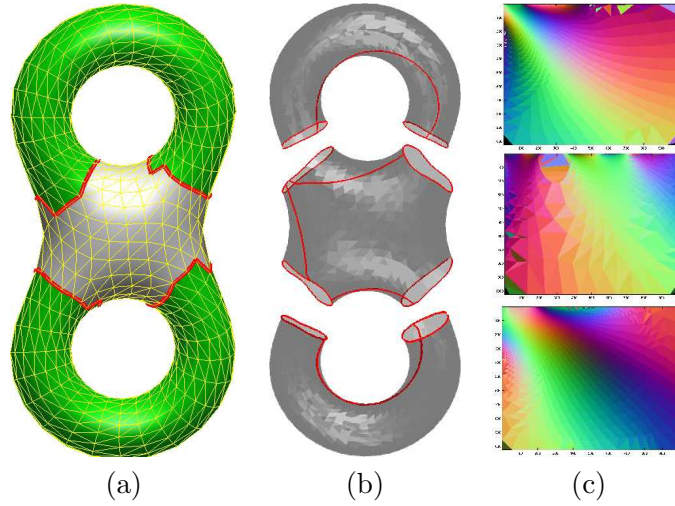


may be drawn as a chain of connected arcs, branches are sorted according to their importance into the contour tree and progressively rendered. The representation of the tree is embedded into the 3D space by moving each node to a  $z$  value that corresponds to the value of the scalar field in that node. In addition, 2D self-intersections are avoided using the algorithm for rooted trees proposed in [dBETT99]. More recently, these branches have been used to topologically simplify the rendering of complex volumetric data sets [CSvdP04, WDC<sup>+</sup>07].

In the field of image processing, contour trees [MM05, Tur98] and their variations [Jon99, MG00, SOG98, BHH98] have been mainly used for optimizing the coding and the manipulation of images and their meaningful components. While the topology of image surfaces is simple (an image may be easily represented as a scalar field whose scalar function is provided by the gray-level intensity), the configuration of contour trees provides an interesting support for many filtering and segmentation operations. In particular, the max tree has been adopted in a number of Computer Vision problems including stereo matching, image filtering, segmentation and information retrieval. First introduced in Computer Graphics by Shinagawa et al. [SKK91], Reeb graphs were initially used only for Morse mapping functions and their extraction required *a priori* knowledge of the object genus [SK91].

Application fields related to the use of Reeb graphs include surface analysis and understanding [SKK91, ABS03]; identification of topological quadrangulations [HA03]; data simplification [BFS02]; animation [KS00, XSW03]; surface parameterization [SF01, PSF04, ZMT05] (Figure 2.14) and remeshing [WDSB00]; shape [BDP06] and human body [WXS06] segmentation; approximation and modification of the input geometry [SK91]; object reconstruction [BMS00] and editing where the stored information is exploited for shape recovering. Moreover, the knowledge of the shape topology given by the graph structure improves the surface reconstruction from contour lines [BMS00], thus solving the correspondence and the branching problems. Details on this topic may be found in [FKU77, MSS92, OPC96].

The compactness of the one-dimensional structure, the natural link between the function and the shape, and the possibility of adopting different functions for describing different aspect of shapes have led to a massive use of Reeb graphs for similarity evaluation, shape matching and retrieval [HSKK01, BMM<sup>+</sup>03a]. In [HSKK01] the Reeb graph is used in a multi-resolution fashion for shape matching. The ratio of the area and the length of the model sub-part in the whole model are associated with each node, i.e. shape slice, and are used as attributes during the graph-matching phase. The set of the geometric attributes is further enriched in [TS05], where, for each slice, the authors consider the volume, a statistic measure of the extent and the orientation of the triangles, an histogram of the Koenderink shape index, which provides a representation of the local shape curvatures [Koe90], and a statistic of the texture. In particular, [TS04] shows how the performance for shape retrieval improves when these geometric attributes are added to nodes of the multi-resolution graph structure. Finally, the method in [HSKK01] has been successfully applied also to database retrieval of CAD models as proposed in [BRS03].



**Figure 2.14:** Surface parameterization using the Extended Reeb Graph. (a,b) A topology based decomposition of the shape derived from the ERG is used to define a chart decomposition of the mesh, and each chart is parameterized with respect to the cuts shown in (b); (c) the normal-map images.

In the field of regularly sampled 3D grids of scalar values (that is a volume model in which each grid cube has 8 neighbor grid points), the method proposed in [WHDS04] topologically simplifies and repairs the topological noise that affects large scattered datasets. Similarly, the method in [SL01] analyzes, and eventually corrects, the topology of cortical volumes because it is assumed that cortical volumes are homeomorphic to a sphere and no cycles are expected in the graph representation.

Similarly, the method proposed in [PSBM07] has been used to find and highlight small defects, such as small manifold handles and tunnels, in models of arbitrary dimension. Since these features correspond to loops of the Reeb graph, the model is simplified by removing parts that correspond to irrelevant loops, where the relevance of a loop is defined as its persistence (cf. Section 2.3.2), in the sense of a percentage of the function image.

The hyper Reeb graph defined in [FAT99, FTAT00] has been proposed to enhance traditional volume visualization techniques with a double-layered topological structure. In [FTAT00], the method is tested on a large-scale, time-varying volume data set. There, the hyper Reeb graph is used to simulate an ion-atom collision problem between a proton and a hydrogen atom and to investigate the variation of the electron density distribution during the collision. In particular, the identification of the Reeb graphs that correspond to the simplest structure of the isosurfaces permits to approximate the collision time of the atomic structures.

## 2.3 Algebraic shape descriptors

Besides the possibility of adopting different functions for describing shapes, at a higher level of abstraction, the modularity of approaches based on Morse theory can be extended to the choice of the space used to represent the shape, or phenomenon, under study. The third group of methods reflects this higher degree of modularity, and it is concerned with methods allowing one or more real functions to be defined on spaces associated with the shapes under study. These methods are based on algebraic topology, which is a mathematical tool which useful to detect the number and type of topological features, such as holes, in a space. Size theory and persistent homology theory fall in this last group and are characterized by the possibility of varying the space underlying the shape and the real functions defined on it.

### 2.3.1 Size theory

Size theory has been developed since the beginning of the 1990s in order to provide a geometrical-topological approach to the comparison of shapes (cf. [Fro90, Fro91, VUFF93]). Introductory presentations can be found in [FL99] and [KMM04].

The basic notion behind size theory is the abstraction of the similarity between shapes in terms of the *natural pseudo-distance* between the topological spaces that represent the shapes. Intuitively, two shapes are similar when they exhibit similar properties: this fact can be conceptualized by considering a shape as a pair defined by a topological space *and* a function that measures some properties, which are relevant in a specific context. Then, the similarity between shapes can be expressed by a small variation in the measure of these properties when we move from one shape to the other. In this setting, shapes are similar if there exists a bi-continuous transformation, or, more precisely, a homeomorphism, that preserves the properties conveyed by the functions.

The idea comes from the observation that many groups of transformations, such as isometries or affinities, can be expressed in terms of the preservation of the values of some real functions on topological spaces. For instance, the concept of isometry can be traced back to the preservation of the distance function, and the concept of affinity to the preservation of the affine area. Therefore, it seems natural to consider shapes as spaces equipped with real functions and study the changes of such functions, that is, study how the function values are modified or preserved under the action of homeomorphisms between the spaces. The evaluation of such changes yields a measure to compare shapes.

The formalization of this approach provide the definition of the *natural pseudo-distance*, intuitively defined as the infimum of the variation of the values of the functions when we move from one space to the other through homeomorphisms.

Notice that the most common transformation groups, such as those mentioned above, can

be expressed using the language of the natural pseudo-distance [Fro91]. The natural pseudo-distance actually defines a concept of similarity between shapes. Indeed, in this theoretical setting, two objects have the same shape if they share the same shape properties, expressed by the functions' values, i.e. their natural pseudo-distance vanishes.

### 2.3.1.1 Theoretical aspects

The main idea in size theory is to compare shape properties that are described by real functions defined on topological spaces associated with the “objects” to be studied. This leads to considering *size pairs*  $(S, f)$ , where  $S$  is a topological space and  $f : S \rightarrow \mathbb{R}$  is a continuous *measuring function*.

When two objects  $X$  and  $Y$  must be compared, the first step is to find the “right” set of corresponding properties, i.e. of size pairs  $(S(X), f_X)$ ,  $(S(Y), f_Y)$ . Depending on the problem, one can decide to work directly on  $X$ , so that  $S(X) = X$ , or to compute a derived space  $S(X)$  which is different from  $X$ . For example,  $S(X)$  can be chosen (whenever applicable) to be the Cartesian product  $X^{(n)} = X \times X \times \cdots \times X$ , or the tangent space of  $X$ , or a projection of  $X$  onto a line, or the boundary of  $X$ , or the skeleton of  $X$ , and so on. The measuring functions are meant to give a quantitative description of  $S(X)$ ,  $S(Y)$ . Their choice is driven by the set of properties that one wishes to capture.

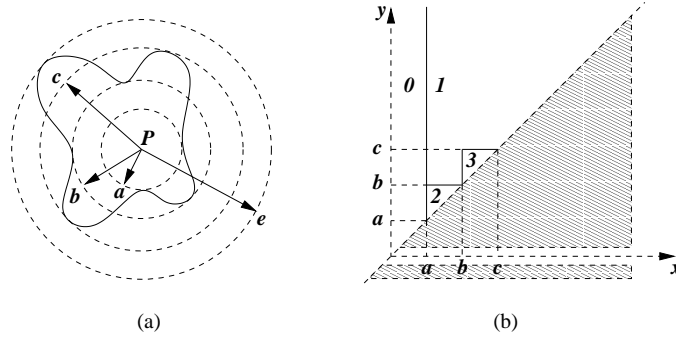
The next step in the comparison process is to consider the *natural pseudo-distance*  $d$ . The main idea in the definition of natural pseudo-distance between size pairs is the minimization of the change in measuring functions due to the application of homeomorphisms between topological spaces. Formally,  $d$  is defined by setting

$$d((S(X), f_X), (S(Y), f_Y)) = \inf_{h \in H_{X,Y}} \sup_{P \in S(X)} |f_X(P) - f_Y(h(P))|,$$

where  $h$  varies in a subset  $H_{X,Y}$  of the set  $H$  of all homeomorphisms between  $S(X)$  and  $S(Y)$ . The subset  $H_{X,Y}$  must satisfy the following axioms: the identity map  $id_X \in H_{X,X}$ ; if  $h \in H_{X,Y}$  then the inverse  $h^{-1} \in H_{Y,X}$ ; if  $h_1 \in H_{X,Y}$  and  $h_2 \in H_{Y,Z}$  then the composition  $h_2 \circ h_1 \in H_{X,Z}$  [Fro91]. Often,  $H_{X,Y}$  coincides with  $H$  (cf. [DF04b, DF07]). If  $S(X)$  and  $S(Y)$  are not homeomorphic the pseudo-distance is set equal to  $\infty$ . It should be noted that the existence of a homeomorphism is not required for  $X$  and  $Y$  but for the associated spaces  $S(X)$  and  $S(Y)$ . In this way, two objects are considered as having the same shape if and only if they share the same shape properties, i.e. the natural pseudo-distance between the associated size pairs vanishes.

Since the set of homeomorphisms between two topological spaces is rarely tractable, simpler mathematical tools are required to estimate the natural pseudo-distance. To this end, the main mathematical tool introduced in size theory is given by size functions, which provide a lower bound for the natural pseudo-distance.





**Figure 2.15:** (a) The size pair  $(S, f)$ , where  $S$  is the curve represented by a continuous line and  $f$  is the function “distance from the point  $P$ ”. (b) The size function of  $(S, f)$ .

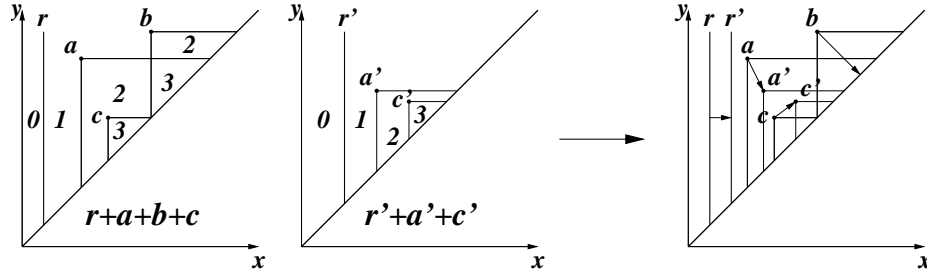
Size functions are shape descriptors that analyze the variations of the number of connected components of the lower level sets of the studied space with respect to the chosen measuring function. Given a size pair  $(S, f)$ , the (reduced) *size function*

$$\ell_{(S,f)} : \{(x, y) \in \mathbb{R}^2 : x < y\} \rightarrow \mathbb{N}$$

can easily be defined when  $S$  is a compact and locally connected Hausdorff space:  $\ell_{(S,f)}(x, y)$  is equal to the number of connected components of the lower level set  $S_y = \{P \in S : f(P) \leq y\}$ , containing at least one point of the lower level set  $S_x$  (see [dFL06]).

An example of size function is illustrated in Figure 2.15. In this example we consider the size pair  $(S, f)$ , where  $S$  is the curve represented by a continuous line in Figure 2.15(a), and  $f$  is the function “distance from the point  $P$ ”. The size function associated with  $(S, f)$  is shown in Figure 2.15(b). Here, the domain of the size function is divided by solid lines, representing the discontinuity points of the size function. These discontinuity points divide the set  $\{(x, y) \in \mathbb{R}^2 : x < y\}$  into regions on which the size function is constant. The value displayed in each region is the value taken by the size function in that region. For instance, for  $a \leq x < b$ , the set  $\{P \in S : f(P) \leq x\}$  has two connected components which are contained in different connected components of  $\{P \in S : f(P) \leq y\}$  when  $x < y < b$ . Therefore,  $\ell_{(S,f)}(x, y) = 2$  for  $a \leq x < b$  and  $x < y < b$ . When  $a \leq x < b$  and  $y \geq b$ , all the connected components of  $\{P \in S : f(P) \leq x\}$  are contained in the same connected component of  $\{P \in S : f(P) \leq y\}$ . Therefore,  $\ell_{(S,f)}(x, y) = 1$  for  $a \leq x < b$  and  $y \geq b$ . When  $b \leq x < c$  and  $y \geq c$ , all of the three connected components of  $\{P \in S : f(P) \leq x\}$  belong to the same connected component of  $\{P \in S : f(P) \leq y\}$ , implying that in this case  $\ell_{(S,f)}(x, y) = 1$ .

In [FL97] a new kind of representation of size functions was introduced, based on the fact that they can always be seen as linear combinations of characteristic functions of triangles (possibly unbounded triangles with vertices at infinity), with a side lying on the diagonal  $\{x = y\}$  and the other sides parallel to the coordinate axes. For example, the size function



**Figure 2.16:** Two size functions can be described by cornerpoints and cornerlines and compared by the matching distance.

of Figure 2.16 (left) is the sum of the characteristic functions of the triangles with right angles at vertices  $a, b, c$  plus the characteristic function of the infinite triangle on the right of line  $r$ . This suggests that the size function is completely determined by  $a, b, c, r$ . In fact, the property that size functions can be represented as collections of vertices (called *cornerpoints*) and lines (called *cornerlines*) always holds [FL01]. This provides a simple and concise representation for size functions in terms of points and lines in  $\mathbb{R}^2$ , drastically reducing the required descriptive dimensionality.

As suggested in Figure 2.16, this representation also allows for the comparison of size functions using distances between sets of points and lines [DFL99], e.g. the Hausdorff metric or the matching distance. In particular, the matching distance between two size functions  $l_1$  and  $l_2$ , respectively represented by the sequences  $(a_i)$  and  $(b_i)$  of cornerpoints and cornerlines, can be defined as

$$d_{\text{match}}(l_1, l_2) := \min_{\sigma} \max_i d(a_i, b_{\sigma(i)})$$

where  $i$  varies in the set of natural numbers  $\mathbb{N}$ ,  $\sigma$  varies among all the bijections from  $\mathbb{N}$  to  $\mathbb{N}$ , and the pseudo-distance  $d$  between two points  $p$  and  $p'$  is equal to the smaller between the cost of moving  $p$  to  $p'$  and the cost of moving  $p$  and  $p'$  onto the diagonal  $\{x = y\}$ , with costs induced by the max-norm [dFL06]. An example is shown in Figure 2.16 (right), where an optimal matching between cornerpoints and cornerlines of two size functions is shown. Here the matching distance is given by the cost of moving the cornerpoint denoted by  $b$  onto the diagonal.

The stability of this representation has been studied in [dFL03, dFL05]. In particular, it has been proven that the matching distance between size functions is continuous with respect to the measuring functions, guaranteeing a property of perturbation robustness. Moreover, it can be shown that the matching distance between size functions produces a sharp lower bound for the natural pseudo-distance between size pairs [DF04a, dFL05], thus guaranteeing a link between the comparison of size functions and the comparison of shapes [dFL06].

Size functions are not the sole tool introduced in size theory. Indeed, algebraic topology has

been used to obtain generalizations of size functions that give a more complete description of a pair  $(S, f)$ , since they take into account not only the number of connected components but also the presence of other features such as holes, tunnels and voids. The first development in this sense can be found in [FM99] where *size homotopy groups* are introduced, inspired by the classical mathematical notion of homotopy group. They are shown to provide a lower bound for the natural pseudo-distance, much in the same way as size functions do.

The study of size functions in the algebraic topology setting was also developed in [CFP01] by observing that  $\ell_{(S,f)}(x, y)$  can be seen as the rank of the image of  $H_0(j_{xy}) : H_0(S_x) \rightarrow H_0(S_y)$  where  $j_{xy}$  is the inclusion of  $S_x$  into  $S_y$ . This observation has led to the definition of the *size functor*, which studies the maps  $H_k(j_{xy}) : H_k(S_x) \rightarrow H_k(S_y)$  for every  $k$ . In other words, it studies the process of the birth and death of homology classes as the lower level set changes. When  $M$  is smooth and compact and  $f$  is a Morse function, the functor can be described by oriented trees, called  *$H_k$  - trees* [CLG01].

### 2.3.1.2 Computational aspects

From the computational point of view, the main efforts have been devoted to the development of techniques for the computation of size functions [Fro92], while no algorithms are available to compute the size homotopy groups or the size functor.

From the application side, using size functions for shape analysis requires two steps: (i) the choice of the size pair  $(S, f)$  and (ii) the computation of  $\ell_{(S,f)}$ .

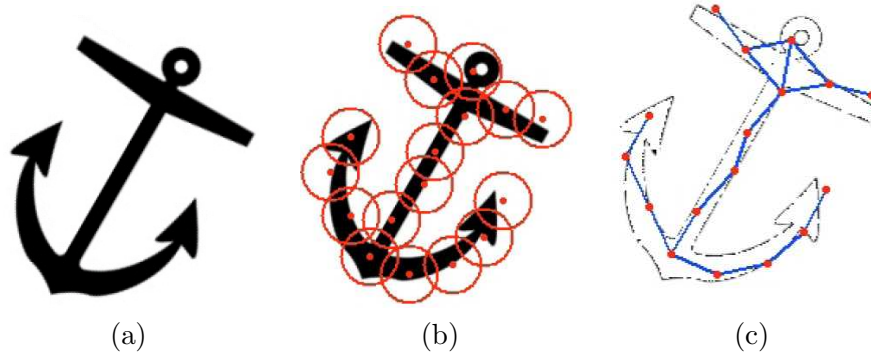
In the following we discuss each of these steps.

### 2.3.1.3 Choosing the size pair

One needs to choose both the space  $S = S(X)$ , associated with the object  $X$  under study, and the measuring function  $f$ . Since each pair  $(S, f)$  conveys information about  $X$  from a different viewpoint, the choice is driven by the set of shape properties that the user wants to capture.

In the applications,  $X$  can be a binary, a gray-scale or a full-color image, a continuous curve, a closed surface or a simplicial complex (in particular a graph). In [UV97] the concept of auxiliary manifold  $S(X)$  is introduced, i.e. an object of fixed and known topological structure that can be linked to the shape  $X$  under study. The definition of the measuring functions on the auxiliary manifold permits the solution of problems related to small topological changes in the original shape due to noise and perturbations. The idea of an auxiliary space as a domain for the measuring functions has also been proposed in the case of binary images in [CFG06] and of surface meshes in [BGSF06].

In [VU96] the authors discuss the need for heuristic criteria to select adequate measuring



**Figure 2.17:** The discretization of a space described in [Fro92]. (a) The original space. (b) The process of  $\delta$ -covering. (c) The approximating graph.

functions, and propose using parameterized families of measuring functions. The key problem of the invariance requirements in Pattern Recognition tasks is explored in [LF02], [VU94] and [DFP04], where the ability of size functions to deal with Euclidean, affine and projective invariance is discussed.

#### 2.3.1.4 Computing $\ell_{(S,f)}$

In recent years three algorithms have been proposed to compute the size functions, once the size pair  $(S, f)$  has been chosen.

The first two algorithms [Fro92, Fro96] involve the definition of two labeled graphs designed to discretize the size pair under study. The third algorithm [d'A00] starts with a given labeled graph and directly computes the cornerpoints and cornerlines, which completely describe the size function.

The algorithm in [Fro92] (see also [Fro91]) requires  $S$  to be a compact and arcwise connected subset of  $\mathbb{R}^m$ , and the measuring function  $f$  to be the restriction to  $S$  of a continuous function  $\bar{f} : \mathbb{R}^m \rightarrow \mathbb{R}$ . The first step of this algorithm consists of the discretization of the space  $S$ . For this purpose, the concept of  $\delta$ -covering is introduced, i.e. a collection of open balls  $\{B(P_i, \delta)\}_i$  of radius  $\delta$  and centered at  $P_i$ , whose union contains  $S$ , and such that the intersection between each ball and  $S$  is a non-empty arcwise connected set. We can define the *size graph approximating*  $(S, f)$  as the labeled graph  $(G, \bar{f}|_G)$ , where the set of vertices of  $G$  is equal to  $\{P_i\}_i$ , and two vertices,  $P_i$  and  $P_j$ , are adjacent if the intersection between  $S$  and the union of the balls  $B(P_i, \delta)$  and  $B(P_j, \delta)$  is an arcwise connected set (see Figure 2.17). Once the approximating size graph has been obtained, the size function of  $(G, \bar{f}|_G)$  is computed.

The algorithm proposed in [Fro96] assumes  $S$  to be a compact and smooth manifold without boundary, and the measuring function  $f$  to be Morse. This method is based on the construction of a *Morse graph*, whose vertices correspond to the critical points of  $f$ , and where an arc exists between two vertices if and only if they are connected by at least a gradient flow line. It has been shown that the size function of the Morse graph is identical to the size function of the original space.

Both the preceding algorithms reduce the computation of the size function of the chosen size pair  $(S, f)$  to that of a labeled graph. The approximating size graph and the Morse graph are examples of *size graphs*. In general, a *size graph* is a size pair  $(G, f)$ , where  $G = (V(G), E(G))$  is a finite graph, with  $V(G)$  and  $E(G)$  the set of vertices and edges respectively, and  $f : V(G) \rightarrow \mathbb{R}$  is any function labeling the nodes of the graph. Denoting by  $G_y$  the subgraph of  $G$  obtained by erasing all vertices of  $G$  at which  $f$  takes a value strictly greater than  $y$ , and all edges that connect those vertices to other vertices, the size function  $\ell_{(G,f)}(x, y)$  of  $(G, f)$  is equal to the number of connected components of  $G_y$ , containing at least one vertex of  $G_x$ .

Since the number of vertices and edges of size graphs can be very large, procedures to reduce them are needed. In [FP99] the instruments of *L-reduction* and  *$\Delta$ -reduction* have been introduced. Their main feature is that they permit a simplification of the graph without changing the corresponding size function.

The *L-reduction* is a global reduction method, since it requires knowledge of the entire size graph. Basically, the vertices of the *L-reduced* graph are of two types: vertices corresponding to local minimum values of the measuring function, and vertices corresponding to saddles of lowest elevation between two minima. The edges connect the minima with their corresponding saddles.

The  *$\Delta$ -reduction* is a local method, in the sense that only the knowledge of the local structure of the size graph is needed. Moreover,  *$\Delta$ -reduction* has been extended in [d'A00] to  *$\Delta^*$ -reduction*, a procedure defined by recursively applying three different editing moves. It has been proven that a *total  $\Delta^*$ -reduction* exists, i.e. that the editing process cannot proceed infinitely, and that the totally  *$\Delta^*$ -reduced* graph has the simple structure of a tree (or a forest, i.e. a disjoint union of trees, when  $G$  has several connected components). A possible implementation for this algorithm is based on the union-find structure [TvL84], so that its computational cost is  $O(n + m \cdot \alpha(2m + n, n))$ , where  $m$  and  $n$  are the number of vertices and edges in the graph, respectively, and  $\alpha$  is the inverse of Ackermann's function. Note that  $m$  is  $O(n^2)$  in the worst case.

The value of the  *$\Delta^*$ -reduction* relies not only on reducing the numbers of vertices and edges in the graph, but also on permitting faster computation of size functions [d'A00]. In fact, the  *$\Delta^*$ -reduction* permits direct computation of the cornerpoints and cornerlines, that completely describe the size functions (see Section 2.3.1.1). To achieve this, one has to orient the reduced graph obtained through the process of  *$\Delta^*$ -reduction* by orienting each edge from

the vertex with higher value to the other one. The resulting configuration is an arborescence, i.e. an oriented tree in which no two edges are directed to the same vertex. Finally, the cornerpoints and cornerlines are computed from this arborescence by applying a recursive procedure. The cost of computing the size function of the reduced graph is  $O(n' \log n')$ , with  $n'$  the number of vertices in the reduced graph; usually  $n'$  is considerably smaller than  $n$ .

### 2.3.2 Persistent homology

Persistent homology provides a tool to study topological features of spaces endowed with possibly varying real functions, completely based on algebraic topology. Among the various tools offered by algebraic topology to detect shape features, homology groups have the advantage of being computable. However, standard homology is not able to decide to what extent a topological attribute of a space is relevant for the shape description.

Morse Theory explores the topological attributes of an object in an evolutionary context. In [ELZ00] and [ELZ02] this evolutionary approach has been revisited. The authors introduce a technique, called *persistence*, which grows a space incrementally and analyzes the topological changes that occur during this growth. In particular, they produce a tool, called *persistent homology*, for controlling the placement of topological events (such as the merging of connected components or the filling of holes) within the history of this growth. The aim is to furnish a scale to assess the relevance of topological attributes. The main assumption of persistence is that longevity is equivalent to significance. In other words, a significant topological attribute must have a long life-time in a growing complex. In this way, one is able to distinguish the essential features from the fine details.

Much of the material about persistent homology has been incorporated in [Zom05]. Recently, persistent homology was surveyed in [EH07].

#### 2.3.2.1 Theoretical aspects

The first concept related to persistent homology theory is that of a filtered complex, that is, a complex equipped with a filtration. A *filtration* of a complex is a nested sequence of subcomplexes that ends with the complex itself. Formally, a complex  $K$  is filtered by a filtration  $\{K^i\}_{i=0,\dots,n}$  if  $K^n = K$  and  $K^i$  is a subcomplex of  $K^{i+1}$  for each  $i = 0, \dots, n-1$ . An example of filtered complex is given in Figure 2.18 (Top). Since the sequence of subcomplexes  $K^i$  is nested, one can think of  $K$  as a complex that grows from an initial state  $K_0$  to a final state  $K^n = K$ . Therefore it is often referred to as a growing complex.

Filtered complexes arise naturally in many situations. The simplest example of filtration is the *age filtration* [ELZ02]: The complex  $K$  is filtered by giving an ordering  $\Delta_0, \Delta_1, \dots, \Delta_m$  to its simplices and by defining the sequence of its subcomplexes  $K^i$  as  $K^i = \{\Delta_j \in K : 0 \leq j \leq i\}$ . In other words the complex grows from  $K^0 = \{\Delta_0\}$  adding each simplex one by

---

one according to the given order. It is assumed that if  $\Delta_i$  is a face of  $\Delta_j$  then  $\Delta_i$  enters the filtration before  $\Delta_j$ .

A filtered complex also arises when some space (e.g. a curve or a surface) is known only through a finite sample  $X$  of its points. Since the knowledge of the original space is necessarily imprecise, a multi-scale approach may be suited to describe the topology of the underlying space (see also the approach in [NSW06] to compute the homology of submanifolds from random samples). The idea is to construct, for a real number  $\epsilon > 0$ , an abstract simplicial complex  $R_\epsilon(X)$ , called the *Rips complex*, whose abstract  $k$ -simplices are exactly the subsets  $\{x_0, x_1, \dots, x_k\}$  of  $X$  such that  $d(x_i, x_j) \leq \epsilon$  for all pairs  $x_i, x_j$  with  $0 \leq i, j \leq k$ . Whenever  $\epsilon < \epsilon'$ , there is an inclusion  $R_\epsilon(X) \rightarrow R_{\epsilon'}(X)$  that reveals a growing complex (cf. [CZCG04b], [CZCG04c]).

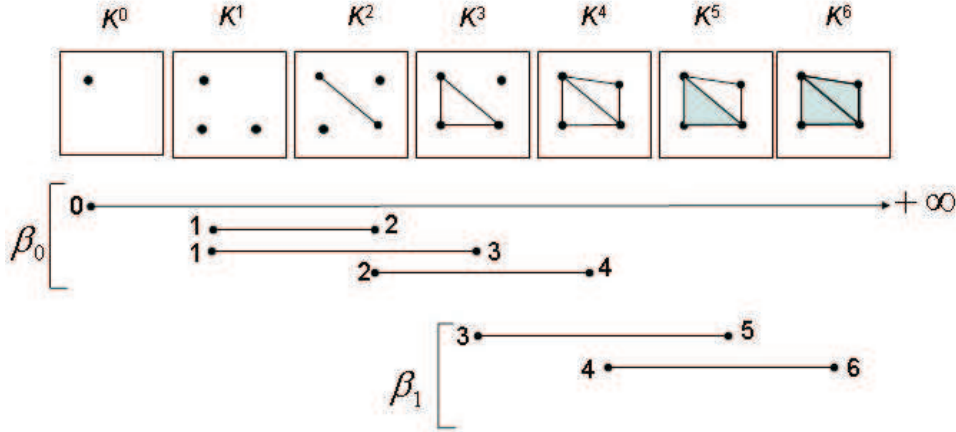
Another example of filtration is provided by a complex filtered by the increasing values of a real piece-wise linear function defined on it. In other words, suppose we are given a complex  $K$  and a piece-wise linear real function  $f$  on  $K$  (that is, a function defined by its values on the vertices of  $K$ ). Denoting by  $A_0, A_1, \dots, A_n$  the vertices of  $K$ , it is possible to filter  $K$  by the subcomplexes  $K^i$  of  $K$ , consisting of the vertices where the function  $f$  takes values not greater than  $f(A_i)$ , together with all the simplices (edges, triangles, etc.) connecting them (cf. [EHZ01], [EHNP03]).

In general, given a filtered complex, its topological attributes change through the filtration, since new components appear or connect to the old ones, tunnels are created and closed off, voids are enclosed and filled in, etc.

In particular, as for 0-homology, each homology class corresponds to a connected component, and a homology class is born when a point is added, forming a new connected component, thus being a 0-cycle. A homology class dies when two points belonging to different connected components, thus belonging to two different 0-cycles, are connected by a 1-chain, thus becoming a boundary. As an example, consider the filtered complex of Figure 2.18 (top): One 0-homology class is born at  $K^0$ , two other homology classes are born at  $K^1$ ; at  $K^2$ , a new homology class is born while one of the classes born at  $K^1$  dies, since it is merged to the class born at  $K^0$ ; at  $K^3$  another class dies, and the same happens at  $K^4$ , where we are left with just one class that survives forever.

As for 1-homology, a homology class is born when a 1-chain is added, forming a 1-cycle (for instance, a 1-simplex is added, completing a circle), while it dies when a 2-chain is added so that the 1-cycle becomes a boundary (for instance, a 2-simplex fills a circle). In the example of Figure 2.18 (top), a homology class is born at  $K^3$ , another one at  $K^4$ , then at  $K^5$  the homology class born at  $K^3$  dies and at  $K^6$  also the homology class born at  $K^4$  dies, so that no 1-cycle survives any longer.

The argument goes on similarly for higher degree homology. Persistent homology algebraically captures this process of the birth and death of homology classes.



**Figure 2.18:** The persistent homology of a filtered complex can be represented by  $\mathcal{P}$ -intervals.

Given a filtered simplicial complex  $\{K^i\}_{i=0,\dots,n}$ , the  $j$ -persistent  $k$ -th homology group of  $K^i$  can be defined as a group isomorphic to the image of the homomorphism  $\eta_k^{i,j} : H_k(K^i) \rightarrow H_k(K^{i+j})$  induced by the inclusion of  $K^i$  into  $K^{i+j}$ . In other words, the  $j$ -persistent homology group of  $K^i$  counts how many homology classes of  $K^i$  still survive in  $K^{i+j}$ . Persistence represents the life-time of cycles in the growing filtration (much in the same way as in the definition of the size functor described in Section 2.3.1.1). Note that, in more recent papers, the notation for persistent homology groups has changed: the variables  $i, j$  have been replaced by  $i, i + j$ .

The persistent homology of a filtered complex can be represented by a set of intervals, called persistence intervals (briefly  $\mathcal{P}$ -intervals), as in Figure 2.18 (bottom). More precisely, a  $\mathcal{P}$ -interval is a pair  $(i, j)$ , with  $i, j \in \mathbb{Z} \cup \{+\infty\}$  and  $0 \leq i < j$ , such that there exists a cycle that is completed at level  $i$  of the filtration and becomes a boundary at level  $j$ .

More recently [CSEH05],  $\mathcal{P}$ -intervals have been described as sets of points in the extended plane. These sets of points are called *persistence diagrams*. In the case of the 0-degree homology of complexes filtered by the lower level sets of a function  $f$  they substantially coincide with the representation of size functions by cornerpoints and cornerlines (cf. Section 2.3.1.1). The difference lies in the assumptions about the space and function allowing for their definition, that in size theory are more general. In the same way that the representation of size functions through cornerpoints and cornerlines satisfies the property of stability under perturbations of the data, so this remains true for persistence diagrams, as proven in [CSEH05, CSEH07b] with respect to Hausdorff and bottleneck distances.

Considering persistence for complexes filtered by the values of a Morse function  $f$ , the process of capturing births and deaths of homology classes naturally establishes a pairing for critical points of  $f$ . For example, when  $f$  is defined on a curve, passing a local minimum creates



a component, while passing a local maximum merges two components represented by two local minima: the maximum is paired with the higher of the two local minima. Clearly, some critical points will not be paired in this process. They correspond to essential features of the shape under study, not depending on the function  $f$ . In [CSEH07a], the persistence pairing is extended to include the homology classes that cannot be paired by ordinary persistent homology since their lifetime is infinite. Another way to pair essential homology classes has been proposed in [DW07], where *interval persistence* is introduced focusing on the stability of critical points rather than critical values. In the case of a function defined on a 2-manifold, the pairing of essential classes can be dealt with also using the Reeb graph [AEHW04] (see also Section 2.3.5).

A recent advance in persistent homology theory is the use of a family of real functions continuously varying in time [CSEM06]. The structures obtained, referred to as *vines* and *vineyards*, seem to be suitable tools for studying continuous processes. Roughly speaking, a vineyard is a bunch of possibly intersecting curves (the vines) generated by the continuous evolution of persistence diagrams over time.

Generalizing persistent homology to the case of a multi-variate situation in which two or more functions are used to characterize the shape leads to the definition of *multi-filtration* [ZC07]. However, it is not possible to extract from this structure a complete and concise representation generalizing the concept of persistence intervals. An alternative solution to this problem is given in [CDF07].

We refer the reader to [Gra03], [Rob99], [Rob02] for different developments of the multi-scale approach to describe the topology of a space, which we do not describe here in more detail, since the filtration of the complex does not vary according to the choice of a real function defined on the space.

### 2.3.2.2 Computational aspects

Applying persistent homology requires the user to model the shape under study with a filtered complex. In each application, the choice of the most suitable filtration is left to the user, much in the same way as the choice of the size pair in size theory.

Once a filtered complex has been obtained, two different algorithms are available to compute persistent homology. Both take as input a filtered complex. However, they greatly differ both in the techniques used and in the scope of applications. The first algorithm reported ([ELZ00]) required the complex to be embedded in  $\mathbb{R}^3$  and filtered by the age filter. Moreover, it worked only with homology coefficients in the field  $\mathbb{Z}_2$  of integers modulo 2. An advantage of this algorithm is that its rationale is rather intuitive. The second algorithm, presented in [ZC04], is much more general since it allows us to compute the persistent homology of any filtered complex with coefficients over an arbitrary field in any dimension. The drawback is that the algebraic machinery needed to obtain this algorithm is rather sophisticated,

although the final procedure is based on Gaussian elimination. Both the algorithms take at most  $O(m^3)$  in time, where  $m$  is the number of simplices in the filtration.

We now describe the first algorithm ([ELZ00, ELZ02]). The idea is to pair the creation of a cycle with its conversion to a boundary. Such a pairing algorithm takes as input a complex  $K = \{\Delta_0, \Delta_1, \dots, \Delta_m\}$  embedded in  $\mathbb{R}^3$  and filtered with the age filtration, and returns a list of simplex pairs  $(\Delta_i, \Delta_j)$ , where  $\Delta_i$  is a  $k$ -simplex and  $\Delta_j$  is a  $(k+1)$ -simplex ( $0 \leq k \leq 2$ ). Each pair represents a  $k$ -cycle created by  $\Delta_i$  and turned into a  $k$ -boundary by  $\Delta_j$ . The algorithm initially needs to decide whether the addition of a  $k$ -simplex creates a  $k$ -cycle (the cycle question). This question can be answered using the incremental algorithm presented in [DE95]. Here the assumption that  $K$  is embedded in  $\mathbb{R}^3$  plays an important role. Indeed, this implies that the only interesting case is  $k = 1$ , because any 0-simplex belongs to a cycle, no 3-simplex belongs to a cycle and the case  $k = 2$  can be reduced to the case  $k = 1$  using a dual graph (that is, the graph whose vertices correspond to the 3-simplices of  $K$  and where there is an edge connecting two vertices if and only if the corresponding 3-simplices share a common face). The algorithm is called incremental because it consists of adding one simplex of  $K$  at a time. A 1-simplex creates a cycle if and only if its two endpoints belong to the same component. This can be decided by implementing a union-find structure [TvL84]. Once the cycle question is decided for each simplex, the idea is to pair simplices as follows: if a  $(k+1)$ -simplex  $\Delta_j$  does not belong to a  $(k+1)$ -cycle, consider its boundary  $d = \partial_{k+1}(\Delta_j)$ . Since  $d$  is a  $k$ -cycle, we can consider all the  $k$ -simplices in  $d$  that have been previously marked as creating a  $k$ -cycle. The youngest one, that is, the one with the largest index is the simplex  $\Delta_i$  that must be paired with  $\Delta_j$ .

The second algorithm that computes persistent homology ([ZC04], [ZC05]) works on any filtered  $d$ -dimensional simplicial complex  $\{K^i\}_{i=0,\dots,n}$  and over any field  $\mathbb{F}$ . Taking the homology coefficients in a field, the homology groups are actually vector spaces. The key observation is that the computation of persistence requires compatible bases for  $H_k(K^i)$  and  $H_k(K^{i+p})$ . The idea underlying the algorithm is to represent the lifetime of a simplex by utilizing multiplication by a formal parameter  $u$ : if a simplex  $\Delta$  enters the filtration at time  $i$ , at time  $i+1$  it becomes  $u \cdot \Delta$ , at time  $i+2$  it becomes  $u^2 \cdot \Delta$ , and so on. We underline that the letter  $u$  is only a formal parameter, while the actual information about the lifetime is given by the power of  $u$ . With this representation, the boundary operator has coefficients in the ring  $\mathbb{F}[u]$  of polynomials over  $\mathbb{F}$  in the letter  $u$ . It follows that the boundary operator can be represented by a matrix with polynomial entries. A standard reduction algorithm on this matrix, more precisely Gaussian elimination by elementary operations on the columns, reduces the matrix to column-echelon form. By non-trivial algebraic arguments, the  $\mathcal{P}$ -intervals for the  $(k-1)$ -th persistent homology can be directly read from the pivots of the boundary operator  $\partial_k$ , reduced into column-echelon form.

Finally, the computation of *vineyards* is carried out in [CSEM06] through an algorithm that maintains an ordering of the simplices in the filtration as time varies. This can be achieved in time  $O(n)$  per transposition of two simplices in the filtration, where  $n$  is the number of

simplices used to represent the topological space.

### 2.3.3 Morse descriptor

The Morse descriptor is an algebraic descriptor for objects modeled as smooth manifolds endowed with a Morse function that measures some metric properties of the given objects. The main difference with respect to persistent homology is the use of relative homology groups instead of ordinary homology. Similarly to other descriptors based on algebraic topology, this descriptor provides information that can remain constant, despite the variability in appearance of objects due to noise, deformation and other distortions. At the same time, it allows for a significant reduction in the amount of data, while providing sufficient information to characterize and represent objects.

#### 2.3.3.1 Theoretical aspects

This descriptor is based on the use of relative homology groups. Roughly speaking, the relative homology groups of a pair of spaces  $(X, A)$ , with  $A \subseteq X$ , count the number of cycles in  $X$ , while ignoring all the chains of  $X$  contained in  $A$ . For the precise definition of relative homology groups we refer the reader to [Spa66].

In [ACZ04], a descriptor is introduced, defined as the function  $R_f : \mathbb{R}^2 \times \mathbb{N} \rightarrow \mathbb{N}$  defined by setting  $R_f(x, y, k)$  equal to the Betti number of the relative homology group  $H_k(M_y, M_x)$ , when  $x \leq y$  and  $y$  belongs to the range of  $f$ , and equal to 0, otherwise. Here  $M$  is assumed to be a connected compact manifold without boundary and  $f : M \rightarrow \mathbb{R}$  a Morse function. The connectedness assumption guarantees that  $f$  ranges over an interval and its compactness ensures that this interval is finite and closed. Moreover, the absence of boundary guarantees that the homology generators are directly related to the critical points of the function. In fact, if the manifold has non-empty boundary, some of the homology generators can be related to the boundary components. The idea under this definition is that the relative homology groups give information about a change in the topology of the lower level set when going through a critical value. Indeed, if a single critical point of index  $\lambda$  is present between two levels,  $x$  and  $y$ , then the relative homology group  $H_k(M_y, M_x)$  has rank equal to 1 if  $k = \lambda$ , and 0 otherwise.

We point out that the intimate connection between the relative homology groups of the lower level sets and critical levels has been known for a long time in mathematics. For example, it was used in [KP47] to define critical levels for any bounded real function  $f$ , defined on any topological space  $X$ , without restricting  $f$  any further.

The idea, sketched in [ACZ04], is more thoroughly developed in [AC07], where the authors distinguish between a *Morse Descriptor (MD)* that corresponds to the function  $R_f$  above, and a *Morse Shape Descriptor (MSD)*, that is a Morse Descriptor defined only for Morse

Algebraic shape descriptors		
Approach	Descriptor	Costs
[d'A00] <sup>8</sup>	size graph	$O(n + m\alpha(2m + n, n))$
[d'A00] <sup>9</sup>	size function	$O(n' \log n')$
[ZC04, ZC05]	persistent homology	$O(N^3)$
[AC07]	Morse shape descriptor	$O(N^2)$

**Table 2.3:** Computational costs of descriptors based on algebraic theory. Symbols:  $n$  represents the number of vertices or points or pixels (voxels);  $m$  corresponds to the number of edges;  $n'$  is the number of vertices of the size graph;  $N$  is the number of simplices or complexes.

functions invariant under rigid motions and scale transformations.

The Morse Descriptor allows for multi-scale analysis of shape, since, as shown in [AC07], the larger the number of the lower level sets studied, the more topological information can be obtained:  $MD_f(x, y, k) \leq MD_f(x, z, k) + MD_f(z, y, k)$ , for every  $x \leq y \leq z$  in  $\mathbb{R}$  and every  $k \in \mathbb{N}$ .

### 2.3.3.2 Computational aspects

In its discrete form, the Morse shape descriptor is encoded in a collection of matrix structures, one matrix for each homology degree. For a manifold of dimension  $n$ , there are exactly  $n + 1$  significant homology degrees  $k$ ,  $0 \leq k \leq n$ . The element in the  $i$ -th row and  $j$ -th column of the  $k$ -matrix is precisely the number  $R_f(x_i, y_j, k)$ . Therefore, the comparison of shapes reduces to a distance measure between matrices. The integer indexes  $i, j$  vary in the set of the first  $N$  natural numbers. The value  $N$  represents the number of samples in the discretization of the range of  $f$  normalized between the absolute minimum and maximum value.

The algorithm presented in [AC07] involves the computation of the relative homology of  $N(N - 1)/2$  pairs of complexes.

In view of applications to images, the algorithm used in [AC07] to compute the homology groups is based on cubical complexes, which are complexes whose basic building-blocks are intervals, squares, cubes and their generalizations to higher dimensions [AMT01], [AZ03], [KMM04], [ZA02].

### 2.3.4 Computational complexity

The computational costs of the extraction of algebraic-based descriptors are summarized in Table 2.3.

### 2.3.5 Applications

The most explored field of application concerning size theory is the field of Pattern Recognition, where size functions have been used as a tool for shape comparison, retrieval and classification, especially in the case of natural or articulated objects. The relationship between the comparison of shapes and the comparison of size functions is studied in [DF04a].

In [VU96] and [UV94] the authors proposed a recognition scheme for the signs in the International Alphabet Sign Language (ISL). A one-parameter family of 72 measuring functions is introduced, together with 72 corresponding feature vectors, to describe the signs belonging to the ISL, represented by the curves defining their outline. The problem of recognizing the sign language by means of size functions has also been addressed in [HZW99], where a pair of moment-based size functions is used as an input to a neural network classifier.

Size functions are also useful in the biomedical field. In [FLP94] a system for the automatic classification of white blood cells, represented by gray-level images, is presented. In this particular case, the choice of a set of adequate and effective measuring functions is driven by the need to take into account the specific morphological features of the leukocyte classes. Similar principles guided the choice of the features in the ADAM (Automatic Data Analysis for Melanoma early detection) project. In [dFS04], [SBC<sup>+</sup>05] size functions and Support Vector Machines are combined to implement an automatic classifier of melanocytic lesions; the system is mainly based on a qualitative assessment of asymmetry, as a parameter to distinguish between nevi and melanomas.

Recently a strategy to address the problem of figurative images was proposed. In [CFG06] a complete system for automatic trademark retrieval based on size functions was implemented, in order to deal with the actual problem of preserving product identity and avoiding copyright infringement. Experiments were performed on a database to retrieve binary trademark images provided by the UK Patent Office.

The problem of image retrieval on the Internet is dealt with in [CFG05]. Ferri and Frosini [FF05] suggest equipping each image on a Web site with a simplified drawing called *keypic* (in alternative to *keyword*). Cerri et al. [CFG05] carry out an experimentation on a set of keypics, thus proposing size functions as a possible ingredient in the solution to the problem of image searching and retrieval on the Internet.

One of the first applications of persistent homology reported in the literature [EHZ01] addresses the problem of topological simplification viewed as a process that decreases Betti numbers. We have seen that a by-product of the computation of persistent homology is a set of pairs of simplexes  $(\Delta_i, \Delta_j)$ , where  $\Delta_i$  is the simplex that, on entering the filtration, creates a cycle, and  $\Delta_j$  is the simplex that, entering the filtration, adds the cycle to the group of boundaries. In other words, attaching  $\Delta_i$  corresponds to the birth of a new ho-

---

<sup>8</sup>The number of edges of the size graph may be  $O(n^2)$  in the worst case.

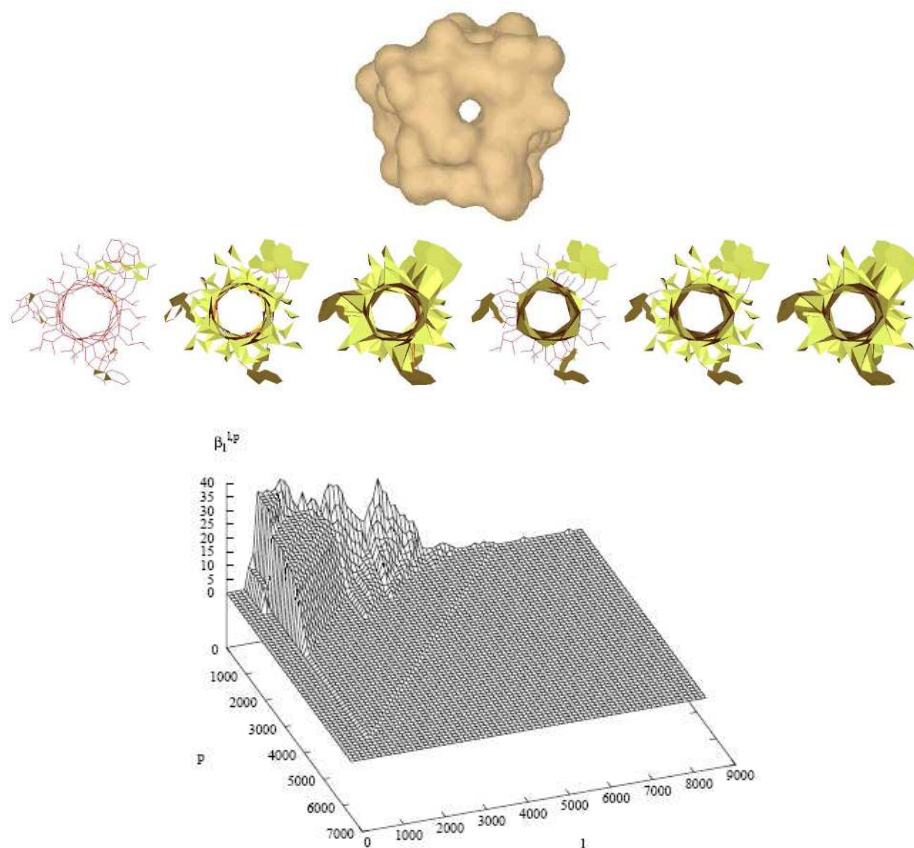
<sup>9</sup>In general the relation  $n' \ll n$  holds.

mology class, while attaching  $\Delta_j$  leads to its death. Thus, topological simplification can be achieved by removing simplexes in pairs, in the order of increasing importance: cycles whose persistence is below some threshold can be removed, since they correspond to noise or non-relevant features; only cycles with a longer life-time are considered important. However, the meaning of the simplification changes according to the context. Topological simplification of a complex filtered by a Morse function corresponds to geometric smoothing of the Morse function. The application of this topological simplification to complexes generated by sample points provides a method for noise reduction in sample data. An example of this simplification process can be found in [ELZ02], carried out on a molecular structure represented as a complex endowed with an age filtration (see Figure 2.19). A further application is the simplification of Morse-Smale complexes for 2-manifolds proposed in [EHZ01], where the sequence of cancellation of pairs of critical points is driven by the persistence of the pairs. This approach is also followed in [BEHP04], where an algorithm is proposed which allows the simultaneous application of independent cancellations (see also the formalization in [DFPV06]). Applications are shown for terrain models. The extension of the idea of the persistence-driven simplification of Morse-Smale complexes to functions defined on 3-manifolds is performed in [GNP<sup>+</sup>05].

The problem of simplifying the function itself, rather than the underlying space, has recently been addressed in [EMP06]. The notion of  $\epsilon$ -*simplification* of a function  $f$  is introduced, that is, a function  $g$  such that  $\|f - g\|_\infty \leq \epsilon$  and whose persistence diagrams are the same as those of  $f$ , except that all points within  $L_1$ -distance less than  $\epsilon$  from the diagonal of  $\mathbb{R}^2$  are removed. It is also shown, through a constructive proof, that  $\epsilon$ -simplifications exist for 2-manifolds. Beside the problem statement, the main novelty in [EMP06] is that, in previous works on simplification, all points of the persistence diagrams could be moved towards the diagonal, regardless of their distance from it, while this approach ensures that points with persistence higher than  $\epsilon$  are not moved. The order of removal of pairs of critical points no longer follows the order of increasing persistence, but the increasing value of the function on the second point of the pair.

A study of shape description and classification via the application of persistent homology is carried out in [CZCG04b, CZCG04c] for curve point cloud data, and in [CZCG04a, CZCG05], where examples are shown for geometric surfaces and surfaces of revolution. The general idea is to describe the shape of a complex  $K$ , filtered by the increasing values of a real function  $f$ , defined on  $K$ : the topological changes occurring through the filtration, that, according to Morse theory, are due to the presence of critical points of  $f$ , are captured by persistent homology. In particular, in these works the shape of  $X$  is studied by constructing a new complex strictly related to  $X$ : the *tangent complex* of  $X$ , that is the closure of the space of all tangents to all points in  $X$ . The tangent complex contains a large amount of information about the geometry of  $X$ . So far, the authors have confined themselves to considering only one function, that is, the curvature at a given point along a specified tangent direction. The choice of this function is directed at capturing those features of a shape that are connected

---



**Figure 2.19:** (a) Top view of the molecular surface for Gramicidin A, presenting a channel as primary topological feature. (b) Filtration of a complex representing the molecular data. (c) Graph of  $\beta_1^{l,p}$  showing that eliminating 1-cycles with low between the feature of the representation. (Provided Zomorodian.)

with curvature. From this setup the authors derive the notion of the *barcode* of a shape, that is, the set of  $\mathcal{P}$ -intervals for this filtered tangent complex. A pseudo-metric between barcodes allows for a measure of the similarity between shapes: being a pseudo-metric and not a metric, the distance between two different shapes can be vanishing. It is interesting to note that, analogous to what happens in size theory, in this research the shape of  $X$  is studied by constructing a new complex, strictly related to  $X$ , which in this case is the filtered tangent complex. Barcodes as descriptors for shape classification have been tested on a small database (80 items) of hand-drawn copies of letters.

The recently introduced concept of vineyards is applied in [CSEM06] to the study of protein folding trajectories. The analysis of the behaviour of proteins also motivates the work in [AEHW04]. The notion of *elevation* is introduced for points on 2-manifolds smoothly embedded in  $\mathbb{R}^3$ , in order to identify cavities and protrusions in the protein structure, since they play an important role in protein interaction. The definition of elevation is derived from an extension of the classical notion of persistence pairing, which takes into account the pairings between all critical points of the function defined on a genus  $g$  embedded 2-manifold. In particular, the  $2g$  saddles starting the  $2g$  cycles, which remain unpaired once the manifold sweep is complete, are paired making use of the Reeb graph of the manifold. Pairing all critical points allows the elevation of each point to be determined by its persistence, that is, the absolute difference in function values to its paired point. Results of the application of extended persistence pairing to protein docking are presented in [WAB<sup>+</sup>05].

Addressing the problem of localizing topological attributes, in [ZC07] *localized homology* is proposed as a method to find the most local basis of the homology of a given space. The main tool is the Mayer-Vietoris blowup complex associated with an open covering of the simplicial complex under study. The blowup complex and the original complex have the same homology but the former can be filtered according to the number of open sets that cover each simplex. The persistent homology of this filtered complex gives insights into the relationship between the local and global homology of the space.

Finally, we point out that the idea of considering the topology of a space at various resolutions by means of persistent homology is inspiring research in new directions, such as the coverage problem in sensor networks [dSG07] and the analysis of the structure of natural images [dSC04].

The performance of the Morse Shape Descriptor has been evaluated in [AC07] on a 2D image retrieval problem. The experimental dataset contains 1100 2D images clustered in 10 classes. In order to model the shapes as connected compact manifolds without boundary, only the shape contours are considered. For each contour, four Morse Shape Descriptors are produced, associated with four different measuring functions, invariant with respect to rigid motions and scale transformations. The measure of similarity between two shapes is given by a weighted sum of distances between the collection of Morse Shape Descriptors associated with the contours. The effectiveness of the system is evaluated by measuring its precision



and recall.

## 2.4 Shape descriptors based on distance transform

Methods based on Morse theory does not cover the whole spectrum of shape analysis methods. Also, there are a number of shape descriptors that are not directly defined using Morse theory, but closely related to it, at least from the point of view of the extraction procedure. A notable example is the class of methods that use the *distance function* for shape analysis [JBS06]. In this case, the function is actually not applied to the shape itself, but used to define a scalar field of distances between points in space and the shape itself. Similarly, the critical points of the distance field [BDGJ07] are related to the main shape features and define the so-called *medial axis*, probably among the best-known shape descriptors. A recent survey of techniques based on the distance field appeared in [JBS06], and the theoretical properties of the medial axis have been the topic of several papers in the literature [Aur91, Wol92, GK04]. Distance functions are not parametric with respect to the mapping function  $f$  and, differently from the other approaches discussed in the Chapter, methods that use the distance functions provide one single type of description, where the features are those characterized by the distance field.

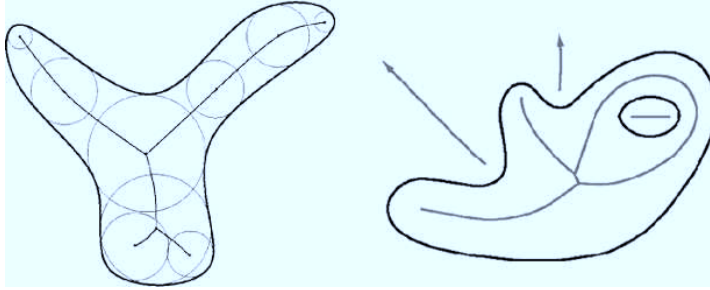
### 2.4.1 Medial axis

The *medial axis transform* (MAT) has been introduced by Blum [Blu67] as a tool in image analysis. To get an intuitive feeling for this concept, consider starting a grass fire along a curve in the plane. The fire starts at the same time, everywhere along the curve, and it grows at constant speed in every direction. The medial axis is the set of locations where the front of the fire meets itself. Formally, let  $X$  be a bounded open subset of the Euclidean  $k$ -dimensional space,  $\mathbb{R}^k$ . The *medial axis*,  $\mathcal{M}[X]$ , is the set of points that have at least two closest points in the complement of  $X$  [Lie03], see Figure 2.20.

#### 2.4.1.1 Theoretical aspects

The medial axis of a shape captures its connectivity, ignoring local dimensionality. More precisely, a shape and its medial axis are homotopy equivalent [Lie03, SPW96, Wol92]. In  $\mathbb{R}^k$ , the medial axis has generically dimension  $k - 1$ , one less than the dimension of the space. In the plane, the medial axis is a (one-dimensional) graph whose branches correspond to regions of the shape it represents. The MAT of planar polygons consists of straight lines and parabolic arcs; each convex vertex of the polygon has an edge of the MAT terminating in it. The MAT structure is very sensitive to noise: the insertion of a new vertex in the boundary of the shape will cause new edges to appear in the skeleton. In  $\mathbb{R}^3$ , it is composed

---



**Figure 2.20:** Medial axis of two planar shapes. In the second example the medial axis is shown also for the external part of the shape.

of pieces of surfaces, and is sometimes called a *medial surface*. When each point  $x$  of the medial axis is weighted with the radius  $\rho(x)$  of the maximal ball centered at  $x$ , then we have enough information to reconstruct the shape. In other words, the medial axis together with the map  $\rho$  provides a reversible coding of shapes. This coding is not necessarily minimal and some shapes, such as finite union of balls, can be reconstructed from proper subsets of their weighted medial axes.

#### 2.4.1.2 Computational aspects

The exact computation of the medial axis is extremely complex in the domain of freeform shapes. In fact, the exact MAT computation was considered for long time affordable only for polygons [Lee82, For87], and more recently for polyhedra [SPB96, CKM04]. Recently, a few researchers have tackled the problem in the context of freeform (piecewise) rational entities.

Today's accepted approach for computing the planar arrangements of freeform geometry approximates the geometry using piecewise lines and arcs, but this method has noteworthy disadvantages. First, the approach is only an approximation. Second, it is also erroneous. The MAT of a planar shape enclosed by two concentric circles is another mean circle in between them. Yet, by tessellating the two input circles into lines, one introduces numerous  $C^1$  discontinuities along these circles. The resulting MAT will consist of numerous and erroneous edges from the mean circle toward all the  $C^1$  discontinuities in the two boundary circles.

As noted above, the construction of the Voronoi diagram and MAT for freeform curves in the plane is more difficult because of the complexity of the bisectors. Ramamurthy and Farouki [FR99a, FR99b] implemented an incremental algorithm in which the bisectors are inserted one by one and the Voronoi diagram of the curves is updated after each insertion; the MAT is derived from the Voronoi diagram and is represented as a piecewise linear approximation of the actual bisector, computed as the envelope of the point-curve rational

bisectors. Ramanathan and Gurumoorthy [RG02] implemented a different tracing algorithm for the construction of the MAT of a freeform shape. This implementation also approximates the edges of the MAT by computing samples of bisector points on the edges and interpolating these sample points. Piecewise linear curves involve the comparison of expressions with two nested square roots [Bur96]. Efficient and fully robust implementations are few [Hel01]. An exact algorithm for not-necessarily convex polyhedra in  $\mathbb{R}^3$  can be found in [Cul00].

A fairly general class of shapes for which it is possible, in principle, to compute the medial axis exactly are the *semi-algebraic sets*. These sets are the solutions of a finite system of algebraic equations and inequalities. The medial axis of such a set is itself semi-algebraic and can be computed with tools from computer algebra.

Conversely, many approaches have been adopted to implement Blum's original definition in the discrete case. Basically, we can distinguish them into four categories, depending on the adopted skeletonisation method: *skeleton extraction from Voronoi diagrams*; *simulation of the grassfire*; *topological thinning*; *skeleton extraction from distance maps*. The medial axis of a planar curve can be thought of as the Voronoi diagram generalized to an infinite set of points (the boundary points) [ACK01, Ogn94a, OK95]. It has been formally shown [BA92] that the Voronoi diagram becomes an increasingly precise approximation of the continuous medial axis as the density of boundary samples increases. Algorithms which actually try to implement the grassfire process are quite rare; examples are the straight skeleton, first introduced by [AAAG95], and the linear axis [TV04a]. Thinning and distance map computation can be directly applied to volumetric discrete representations that are widely used especially in medical applications: most acquisition techniques produce in fact voxel grids, like the Computed Tomography or the Magnetic Resonance Imaging.

A more detailed analysis of medial axis extraction and skeleton computation can be found in [AdB96] for objects in the two-dimensional space and in [dBN05] for the three-dimensional case. Other recent contributions on this topic are provided in [CSM05, BAB<sup>+</sup>07, SP07].

## 2.4.2 Shock graph

Another medial structure is the *shock graph*, [KTZ95], which is obtained by viewing the medial axis as the locus of singularities (shocks) generated during the fire front propagation from the shape boundary.

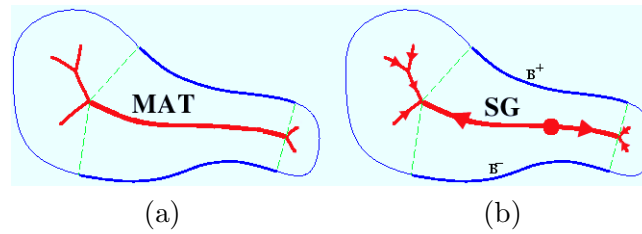
### 2.4.2.1 Theoretical aspects

The shock graph represents a dynamic view of the medial axis associates a direction and an instantaneous speed of flow to each shock point, [GK03]. In particular, shock points may be classified according to the number of contact points and to the flow direction, as described in [GK00]: *source* and *sink* points determine the nodes of the graph while the *links* connect

---

source points to sink ones and define the arcs of the graph. In addition, attributes are associated to the shock graph to store both the intrinsic geometry of the portion of shape corresponding to a link and the radius and the flow direction of each node. Analogously to the MAT, the shock graph structure and the corresponding point classification have been extended to 3D shapes [GK03]. Also, in this case the shock graph structure contains dimensionally heterogeneous components and it is not a planar graph.

The medial axis and the shock graph differ for the interpretation of the structure entities rather than for the geometric abstraction they provide. For example, the shock graph and the MAT of a curve have the same arcs and nodes, but the shock graph associates also to each arc the growing direction of the radius of the bi-tangent spheres, see Figure 2.21(b). In general, we may consider that the shock graph is a finer partition of the medial axis.



**Figure 2.21:** The medial axis (a) and the shock graph (b) of two simple curves.

#### 2.4.2.2 Computational aspects

Shock graphs are inspired to the original idea of Blum [Blu73]. In particular, a shock graph is an abstraction of the skeleton of a shape onto a directed acyclic graph [KTZ95]. Therefore, a quite standard strategy for constructing the shock graph is to extract the skeleton of the shape and then to label and group skeleton points. The skeletonization may be accomplished in several ways; standard approaches are based on Voronoi techniques [Ogn94a, ACK01], Euclidean distance transform [Bor96], topological thinning [Ros98, DPS00] and mathematical morphology [dR03]. For example, in [DPS00] a dynamic programming algorithm is proposed to find the locations of skeleton points in defined by singular values of the divergence equation that also defines shocks. In the definition of shock graphs, shock labels are determined by the radius function in a local neighbourhood of each point [SSDZ98]. Unfortunately, in the case of discrete skeletons, the location of shock points is very sensitive to the accuracy with which both the radii and centers of the locus of the maximal inscribed circles can be computed. Clearly, perturbations to the shape boundary will affect the exact pixel location of the shock, regardless the algorithm used to compute the skeleton. Furthermore, due to the discrete nature of images, the skeletonization algorithms may suffer from instabilities.

The method for the construction of the shock graph proposed in [Mac03] considers as starting point of the algorithm the set of points of the medial axis. In particular, it is supposed such

a set is continuous and connected, where two points are said to be connected if they can be joined by a continuous path of points also in the shock graph. Then the points lying on the skeleton are grouped according to their shock label (i.e. source, sink, branch points, etc.). To obtain a labeling robust to small perturbations of the boundary and to noise in the location of points, it is proposed of finding a small number of line segments to approximate the radius function for each skeleton branch. When shock points have been grouped, the orientation of the skeleton segments is determined according to the value of the radius function. In particular, the nodes of the graph are ordered so that they preserve their adjacency and satisfy the shock graph grammar, while edges are directed from nodes with greater radii to those with smaller radii. Such operations are guaranteed by the results in [KTZ95] where it has been proven that such a ordering exists and it is unique for any closed curve. Finally, a root node is added and edges are inserted to connect the root node with all graph nodes with in-degree equal to zero.

### 2.4.3 Centrelines and Curve skeleton

Centerline skeletons play a relevant role in Computer Graphics and Vision, because they represent a manner to reduce a complex 3D shape to a simple one-dimensional geometric abstraction [LLS92, CSM05].

In general, centerline skeletons are related to the medial axis in the sense that they yield a shape description that always falls inside the shape and aims to be in the *middle* of the volume enclosed by a surface. In the two-dimensional space, the medial skeleton is a union of arcs and curves and reversibility is almost completely guaranteed, starting from the centreline. In turn, in the three-dimensional space, reversibility is possible only if the so called surface-skeleton, consisting of surfaces and curves, is computed. For solid objects, i.e., objects having no cavities, the surface-skeleton can be furthermore compressed to obtain a linear shape representation. In this case, reversibility is no longer possible. In fact, a large number of centers of maximal balls is unavoidably removed from the surface-skeleton to reduce it to the curves keleton.

Traditional techniques to extract a curve skeleton are based on discrete methods that use distance maps and thinning: in these cases the object is stored as a collection of pixels/voxels and the resulting skeleton is a connected subset of these pixels/voxels [CSM05, BAB<sup>+</sup>07]. Recently, Dey et al. [DS06] provided a formal definition of curve-skeletons, which is valid for surfaces without boundary, on the basis of the distance function from a volume boundary.

#### 2.4.3.1 Theoretical aspects

The curve-skeleton [DS06] is defined as the set of points of the medial axis that are singular with respect to a medial geodesic function. More formally, given a space  $O \subset \mathbb{R}^3$  bounded by

a connected manifold surface  $M$ , let  $MA$ ,  $MA \subset O$ , be the medial axis of  $O$ . Two subsets of  $MA$  are defined, namely  $MA_2$  and  $MA_3$ .  $MA_2$  is the set of points whose maximal inscribed balls touch the surface  $M$  at exactly two distinct points.  $MA_3$  is constituted by curves lying at the intersection of the closure of three sheets in  $MA_2$ . The maximum balls lying on points of  $MA_3$  touch the surface  $M$  at three points. A *Medial Geodesic Function (MGF)* is defined on  $MA_2$  and  $MA_3$ . At a point  $x \in MA_2$ , MGF is the length of the geodesic path between two points at the intersection of the surface  $M$  with the maximum ball centered in  $x$ . The definition of MGF is extended to each point of  $MA_3$  on the basis of the values that MGF assumes in the three half-disks of  $MA_2$  around the point. The medial geodesic function is used to define the curve-skeletons of the two subsets  $MA_2$  and  $MA_3$ . The curve skeleton  $Sk_2$  of  $MA_2$  is the set of singular points of MGF on  $MA_2$ . The curve skeleton  $Sk_3$  of  $MA_3$  is the set of points where the gradient flow of MGF sink into from the three local neighbors. Finally, the curve-skeleton of the space  $O$  is defined as the closure of  $Sk_2 \cup Sk_3$ . Since the exact computation of the curve-skeleton is extremely hard, it is approximated by adopting a rough evaluation of the function MGF on the centers of the medial axis facets and a polygonal approximation of the medial axis.

The curve-skeleton and, in general, centerlines related to the concept of medial axis provide a shape description which is unique and independent of other functions defined on the shape. Therefore, these descriptions yield a geometric centerline of the shape rather than a topological one.

#### 2.4.3.2 Computational aspects

The curve-skeleton in [DS06] is obtained by eroding the medial axis. In fact, this implementation of the curve skeleton is based on the approximation of the medial axis based Voronoi diagram filtration proposed in [DZ03]. This approach filters the Voronoi diagram of a set of vertices on the surface and retains a set of Voronoi facets to approximate the medial axis. The main problem with this medial axis computation is that the filtration, often guided by some input parameters, leave some unwanted spikes or holes in the approximate medial axis. To avoid these shortcomings all the Voronoi facets inside the space bounded by the object are kept as a preliminary approximation of the medial axis. Then the shortest geodesic distance on the shape boundary is computed using the algorithm proposed in [SSK<sup>+</sup>05]. The gradient of MGF for any point inside a Voronoi facet is approximated by computing the tangent directions of the shortest geodesic paths and projecting these vectors in the Voronoi facets. A user parameter  $\theta$  is used to mark the vertices having high negative divergence of the gradient field of MGF that must be preserved in the curve skeleton. An erosion process guided by the MGF gradually erodes and collapses edges and vertices of the Voronoi diagram until only significant points, i.e. those with negative divergence, are preserved in the curve skeleton. As the user parameter  $\theta$  decreases the arcs of the curve skeleton corresponding to the less prominent features are removed; if  $\theta < 1$  the curve skeleton reaches the simplest

---

Skeleton extraction methods		
Approach	Description	Costs
[Cga]	Voronoi graph	$O(n^{\lceil \frac{k}{2} \rceil}) + n \log n$
[CSY97]	Voronoi graph	$O((n + F) \log^2 F)^9$
[Lee82]	Medial axis of a polygon	$O(n \log n)$
[CKM04]	Medial axis of a polyhedron	$O(n^{3+\epsilon})^{10}$
	Discrete skeleton 2D images	$O(n^2)$
	Discrete skeleton 3D images	$O(n^3)$
[AAAG95]	Straight skeleton	$O(nr \log n)$
[EE99]	Straight skeleton	$O(n^{1+\epsilon} + n^{8/11+\epsilon} r^{9/11+\epsilon})$
[CV02]	Straight skeleton	$O(n \log^2 n + r \sqrt{r} \log r)$
[TV04a]	Linear Axis	$O(n)$
[DS06]	Curve skeleton	$O(n^2)$

**Table 2.4:** Classification of the methods for skeleton extraction. Symbols:  $n$  represent the number of vertices or points or pixels (voxels);  $m$  the number of vertices inserted in the mesh during an eventual contouring phase;  $e$  the number of edges in the neighborhood tree,  $r$  is the number of reflex vertices.

from consisting of loops only [DLS07].

On the contrary the extraction of curve skeleton proposed in [CSYB05] is based on potential field, adopting a threshold parameter that stops the thinning process of a voxelization of the original triangle mesh.

#### 2.4.4 Computational complexity

As far as computational issues are concerned, in table 2.4 we briefly summarize the complexity of the algorithms for skeleton extraction.

.

#### 2.4.5 Applications

Being the MAT unfortunately hard to be computed in the general case and unstable to small perturbations of the shape, a large number of variations of the MAT were introduced: some of them are just approximations of the MAT for facilitating the skeleton computation (e.g. MAT computation through Voronoi diagrams), while others come from different definitions

<sup>9</sup>For point sets with provable small Voronoi graph,  $F$  is reasonably small:  $F = O(n \log n)$  [ABL03] or  $F = O(n)$  [AB04].

<sup>10</sup>This cost is the best upper bound of the computation of the medial axis and holds for any  $\epsilon$  positive

and present different properties. A few descriptors are able to represent the exact medial axis for a small category of input shapes, like the bisectors of parametric curves and surfaces.

Medial axis and skeleton approximations based on distance transform have been widely used in several application tasks including mesh generation [SERB99], shape representation and description [SAR96], animation [GHK99], image processing [Ogn94b, PGJA03], surface reconstruction [AK00] and solid modeling [STG<sup>+</sup>97]. In particular, the utility of the medial structure has been demonstrate for addressing issues such as: the generation of hierarchical representations that remove local features without loosing the global shape appearance; the generation of regular meshes suitable for the finite element analysis; the object metamorphosis generating a sequence of shapes that transform a shape into another and the generation of a shape from a number of constraints [STG<sup>+</sup>97]. Finally, skeleton-like structures are essential for implicit model animation; in fact, during animation, its attributes may change, varying, for instance, radius, blending and other surface details.

Shock graphs are widely used for image matching (see Figure 2.22(a)), recognition and curve alignment, therefore methods proposed in literature mainly address the problem in the bi-dimensional case and the shape is supposed to be a closed curve.

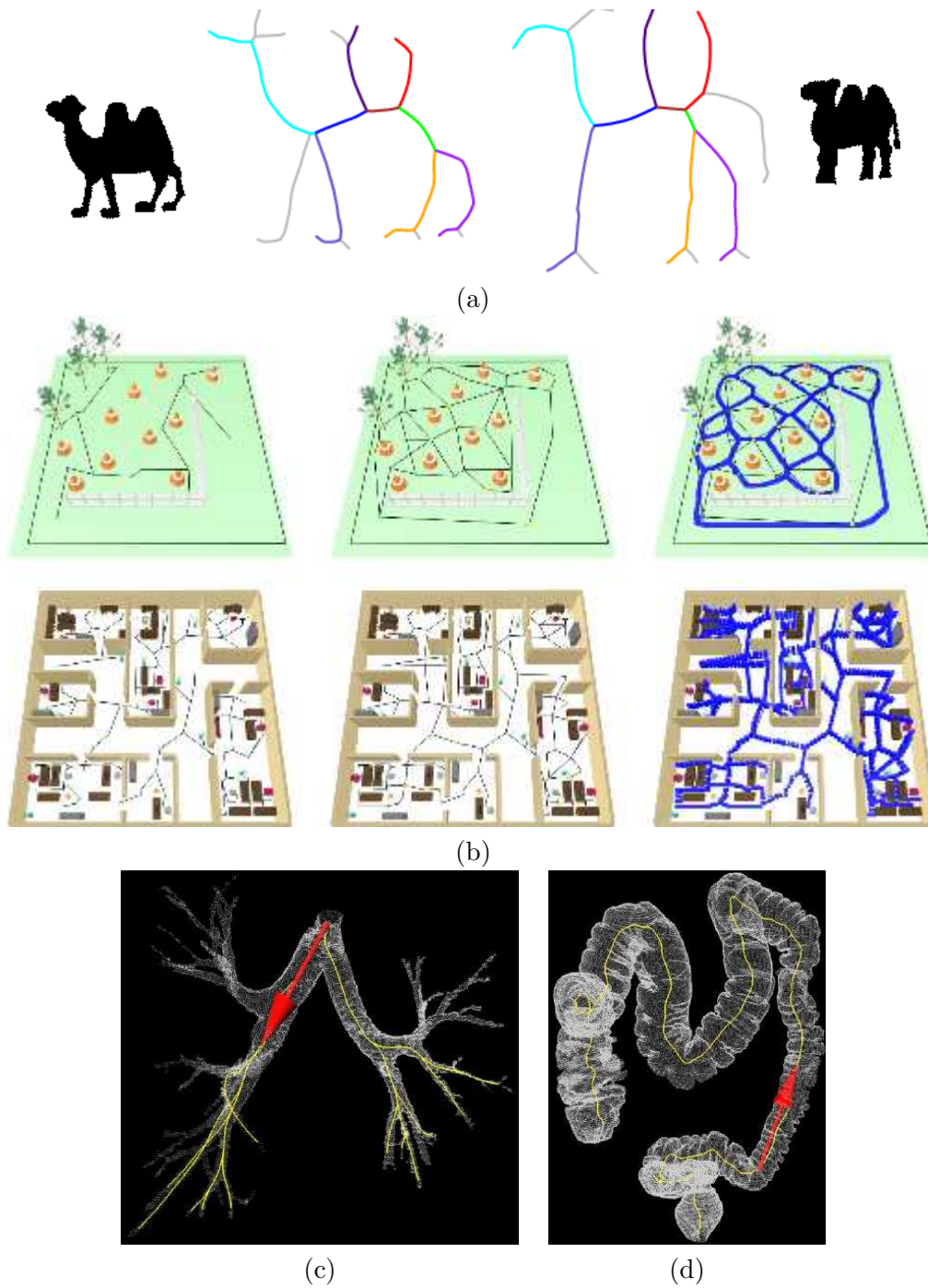
In the 3D case the distinction between the MAT and others skeletal structures becomes more evident: in fact, while the MAT in 3D is essentially a medial surface, in many applications a linear skeleton may be preferable. This is the case of path planning for medical applications in which a linear skeleton, as far as possible from the shape boundary, is needed, maybe to plan the inspection of a human organ [WLK<sup>+</sup>02], see also Figure 2.22(b). Similarly approximate skeleton find application for motion planning in virtual environment contexts, see Figure 2.22(c,d). The definitions and properties of these linear 3D skeletons depend mostly on the input data type: for discrete representations like collection of voxels, distance maps and thinning techniques are used; wave-front propagation and level set approaches are preferred in the continuum case.

Distance transforms are especially suitable for image processing and pattern recognition, and in general for the analysis of discrete objects represented by grids of pixels or voxels. For instance, thinning and distance field computation can be directly applied to volumetric discrete representations that are widely used especially in medical applications: most acquisition techniques produce in fact voxel grids, like the Computed Tomography or the Magnetic Resonance Imaging.

The input to the distance map computation is indeed a grid of discrete points, with each point labeled as being either a feature point or a background point. In this context, feature points correspond to boundary points, and background points to interior ones. The output is the distance transform, i.e. a corresponding grid where each interior point is marked with its relative distance to the nearest boundary point. Note that relative distance may or may be not the Euclidean distance between the points. Some faster and more easily implemented approximations use non Euclidean metrics such as Manhattan distance, chessboard distance,

---





**Figure 2.22:** (a) Image matching using shock graphs. (b) House path planning using approximated skeleton. Cruve skeleton used as central path for virtual bronchoscopy (c) and colonoscopy (d).

and so on.

Distance field based methods works nicely and efficiently for the tubular objects [DS06]. Since the points with the same distances from the boundary of some shapes may form surface patches, the distance field based methods may face difficulty in extracting curve-skeletons for those shapes. Additional applications of the curve skeleton are shape matching, animation and mesh decomposition [CSM05, CSM07].

## 2.5 Discussions and comparative remarks

In this Chapter we have focused on methods for shape analysis in which the object to be studied is described by a pair  $(S, f)$ , where  $S$  is a space, often coinciding with the shape itself, and  $f$  is a real function defined on  $S$ . The shared aim of the methods surveyed is to identify points of interest on a shape, and to capture the shape's connectivity in expressive structures. The approach common to all the methods described finds its roots in classical Morse theory, which combines the topological exploration of  $S$  with quantitative measurement of geometrical properties provided by  $f$ . This means that both global and local analysis concur to obtain the final shape description.

Due to the possibility of adopting different functions for describing shapes, methods based on Morse theory provide a general framework for shape characterization. Changing the properties that one wishes to analyze simply means changing the function, without any other modification in the mathematical model. Moreover, the modularity of the approach can be extended to the choice of the space  $S$  used to represent the shape under study.

We have shown how computational topology enables the solution of questions arising from mathematics and computational geometry. The importance of topology is strengthened to tackle fundamental issues of Computer Graphics such as recognition, deformation, decomposition and homeomorphism. Topology can thus be viewed as a characteristic being more or less in any problem of computational geometry. From this point of view there is an increasing demand of studying the topological questions arising in computer graphics from an algorithmic viewpoint and, among all representation techniques appeared during last years, shape descriptors that organize the shape features in a topological consistent framework are becoming very popular.

In this context we have proposed a brief overview of the main existing topological approaches to shape analysis in the discrete context. In particular, we have focused on those descriptors that provide a high abstraction level representation of the shape structure. We observe that, at the moment, no existing shape descriptor satisfies all “ideal” requirements but, we underline that at the moment all shape descriptors, we have previously discussed, provide a concise representation of the shape that strongly reduces the amount of information stored in the models.

In this Section we propose a global comparison among the techniques described, which highlights the differences in terms of properties of the descriptors (Section 2.5.1), effectiveness of the description and loss of information with respect to the representation (Section 2.5.2), usefulness and context of applicability (Section 2.5.3).

### 2.5.1 Overall comparison and general remarks

The distinguishing feature of Morse and Morse-Smale complexes is that they express geometric information related to the *gradient flow* of the measuring function  $f$ , while the other descriptors encode features captured by  $f$  itself. Contour trees and Reeb graphs compactly represent topological information related to the *level sets* of  $f$ , expressing the way they are connected. Size theory, persistent homology theory and the Morse shape descriptor explore in a homological setting the growth of a space, according to the placement of topological events in the evolution of the *lower level sets*.

We can also observe how the methods surveyed adopt mathematical structures of different level of abstraction to convey geometrical-topological information. On one hand, Morse and Morse-Smale complexes, contour trees and Reeb graphs are essentially combinatoric structures. On the other hand, methods in size theory, persistent homology theory and the Morse shape descriptor mainly rely on the use of algebraic structures.

As for the combinatoric descriptors, the different kind of information they encode is reflected by the differences in the combinatoric structures they produce. While contour trees and Reeb graphs always code the shape in terms of one-dimensional structures (i.e. trees or more general graphs, respectively) disregarding the dimension of the underlying manifold, the decomposition provided by Morse or Morse-Smale complexes is expressed in terms of a cell complex, where the dimension of the complex coincides with the dimension of the manifold. Moreover, Morse and Morse-Smale complexes explicitly induce a shape segmentation. A shape segmentation could also be derived naturally from a Reeb graph, by observing that the counterimages of the simplexes in the graph actually define a decomposition of the shapes into critical level sets and ribbons. The dimension of medial structures may vary, depending on the dimension of the shape domain and the type of description chosen, curve skeleton or medial axis. In the case of skeletons, the domain of the shape coincides with the shape by itself and the distance transform is a tool used to reduce the dimension of the representation. However, it has been shown that a relation exists between the Morse complex of the distance transform and the medial axis of a shape [DS06].

The connectivity of the different complexes is also worth consideration. Ascending and descending Morse complexes are dual to each other. For instance, in a 2-manifold the 0-cells in one complex correspond to the 2-cells in the other complex and vice versa, and there is a one-to-one correspondence between the edges in the two complexes. Thus, by using just one representation for a cell complex, such as the winged edge [Bau75], the half-edge [M88]

or the quad-edge representation [GS85], it is possible to encode the combinatorial structure of both the ascending and descending complexes for a given Morse function  $f$  into a single representation. In this representation, the minima will be associated with the vertices and the maxima with the 2-cells, while the saddle points will be attached to the edges. Two geometric descriptions will be further associated with each edge  $e$ , namely the geometry of  $e$  in the descending and ascending complex, respectively.

In the Morse-Smale complex each vertex corresponds to a critical point and has valence less or equal to four. In particular, in the  $2D$  case, a saddle is always connected with at most two minima and at most two maxima, and a maximum or a minimum is connected with at most four saddles. The 2-cells have four edges and vertices, the latter corresponding to alternating critical points, namely a minimum, a saddle, a maximum and another saddle. These properties allows a description of the critical net of a Morse-Smale complex, called surface network in the  $2D$  case, as a tripartite graph whose nodes are the critical points.

With regard to Reeb graphs, the degree of a vertex, which corresponds to a 0-cell of a Morse-Smale complex, depends on the index of the corresponding critical points. Leaf nodes always represent maxima and minima, and intermediate nodes (i.e., nodes with degree  $\geq 2$ ) correspond to saddles of different index (cf. Section 2.2.2.1). In the case of 2-manifolds embedded in  $\mathbb{R}^3$ , the degree of vertices representing saddles is always three. Similar properties could be stated for contour trees, which are a special case of Reeb graphs. We recall, however, that contour trees are more application-oriented than Reeb graphs, although they are rooted in the same theory.

There are also important differences in the way the shape is coded in the combinatorial representations defined by contour trees and Reeb graphs. Contour trees are defined for scalar fields  $f : D \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$  where  $D$  is a simply-connected sub-domain of  $\mathbb{R}^n$ . This implies that the connected components of the level sets can be ordered with a nesting criteria, and therefore contour trees do not admit cycles. Reeb graphs are defined for a more general class of shapes,  $n$ -dimensional manifolds, and therefore they can have a general connectivity which reflects the topology of the manifold. Moreover, Reeb graphs take into account not only the number but also the changes in the topology of connected components of the level sets, while contour trees do not always (see Section 2.2.2.1). To give an example, let us consider a scalar field in  $\mathbb{R}^3$  whose iso-surface changes genus across a critical value of the scalar field. In this case, the classical contour tree would always count one connected component while the Reeb graph would identify one critical point and therefore reflect its presence in the related graph structure. In other words, the contour tree does not necessarily encode all the saddles that might be identified by analyzing the level set evolution. For all these reasons, Reeb graphs can be thought of as a generalization of contour trees, which allows for a more flexible framework for studying shape properties.

Size theory, persistent homology theory and Morse shape descriptors share a different approach to shape analysis, based on the use of algebraic structures. We remark that an

increase of abstraction level and richness of topological information about the shape corresponds to the increase of mathematical structure – from size functions to persistent homology groups to the size functor – but at the price of diminishing the manageability of the descriptors. In particular, persistent homology groups and size homotopy groups rely on the algebraic notion of group. Since these algebraic structures are difficult to handle, the more manageable tools of persistence diagrams and size functions have been introduced.

The unifying key among size functions, persistent Betti numbers (i.e., the rank of the persistent homology groups) and the Morse shape descriptor is the observation that all these tools basically count some homology classes determined by the inclusion of lower level sets of a space  $S$  with respect to a function  $f$ . Recalling that for  $x < y$  the size function  $\ell_{(S,f)}(x, y)$  counts the number of connected components of the lower level set  $S_x$  which remain disconnected in  $S_y$ , we may notice that this is precisely the rank of the corresponding 0th persistent homology group. This means that size functions actually coincide with the 0th persistent Betti numbers. It also follows that, in the case of 0-degree homology, the formal series describing size function substantially coincide with the set of points of persistence diagrams. The connection between size functions and the Morse shape descriptor is stated in [ACZ04]. An analogous relationship can be stated between the Morse shape descriptor and the rank of the persistent homology group, i.e. the persistent Betti numbers.

It also makes sense to compare persistent homology groups with size homotopy groups since they share the same algebraic structure. In much the same way as the classical result that the first homology group is the abelianization of the fundamental group [Spa66], it can be proved that the first persistent homology group is the abelianization of the first size homotopy group.

Moreover, we observe that the size functor furnishes a shape description by a still more general structure, that is that of functor. In particular, persistent homology groups can be seen as the images of the morphisms of the size functor.

Finally, we highlight that skeletons may consist of a set of geometric primitives such as points, curves, polygons, etc. On the contrary, curve skeletons provides a linear structure, which has the combinatoric structure of a graph. However, the the curve-skeleton extracted using potential eld may not be connected or may not be centered.

### 2.5.2 Expressiveness of shape descriptors

The techniques surveyed can be discussed also from the point of view of their potential for describing as well as for discriminating shapes.

According to the adopted tool, we obtain descriptions which store a different amount of information about the pair  $(S, f)$ . In general, we can observe that the descriptors retaining a larger amount of information are those which allow better discrimination among shapes.

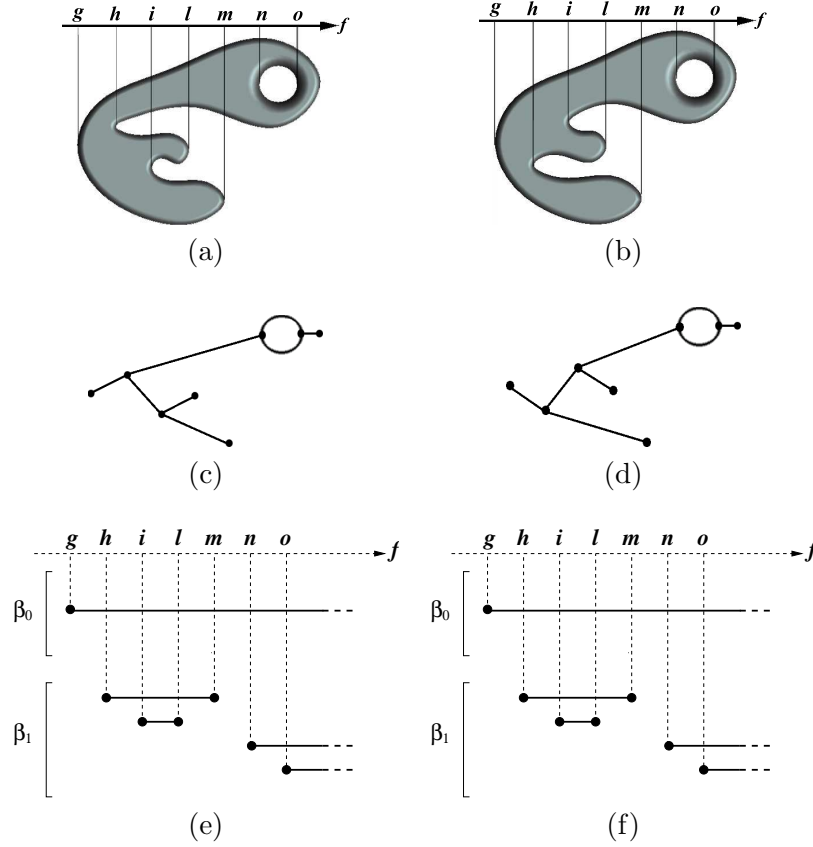
Depending on the application, this can be in turn an advantage or a drawback. The advantage of *forgetting* descriptors is their concise and manageable nature, while their drawback generally is the lack of *completeness*. Completeness here is meant as the property of retaining sufficient data about the shape so that it can be uniquely identified by the descriptor. Conversely, incomplete descriptions might bring up ambiguity in the sense that different shapes may have the same associated representation. These two properties are obviously complementary and the descriptor and abstract representation of the shape has to be devised as a compromise between the two aspects, according to the requirements of the application context.

From a theoretical point of view, the Morse and Morse-Smale complex are complete descriptors among the methods surveyed, since they are naturally associated with a decomposition of the original shape. Notice however that, in applied contexts, it is often better to compute and store the 1-skeleton of the Morse-Smale complex, i.e. the critical net, which is more manageable and computationally less costly. In contrast, in several applications contour trees and Reeb graphs are not treated as purely combinatoric structures, but instead enriched with geometric information related to the underlying model, such as the arc length [WHDS04], some contour levels [SKK91, Bia04a], scalar values related to surface [HSKK01, TS05] or volume [CSvdP04, ZBB04] segments, or the associated subpart decomposition [BMSF06].

From a general perspective, we can observe that there is a decreasing amount of data retained as we progress from combinatoric to algebraic descriptors. For example, consider the 2-manifolds in Figure 2.23. The study of their shape, with respect to the height function  $f$  in the horizontal direction, can be performed using the persistence intervals or the Reeb graph. However, while the Reeb graph is able to discriminate between the two surfaces, neither the 0th nor the 1st homology persistence intervals are able to discriminate between them.

Notice that the triviality of the 0th homology persistence interval is not intrinsic to the manifold, but depends on the particular choice of function. For instance, the opposite of the chosen height function would generate a non-trivial description.

Since the Reeb graph is a one-dimensional simplicial complex with the ability to capture salient shape features, it makes sense to apply other shape descriptors directly to the Reeb graph representation. Since the Reeb graph codes the variation of the connected components of the level sets, it follows that the size functions, the 0th persistent homology group and the 0th homology Morse shape descriptor can be computed on either the original shape or on its Reeb graph. In contrast, this is not true for the size homotopy groups, the higher degree persistent homology groups and the higher degree Morse shape descriptors, since the reduction of the original shape to a one-dimensional structure causes the loss of higher-dimensional features. In particular, the 1st persistent Betti number of the Reeb graph is a lower bound of the 1st persistent Betti number of the original shape, while the subsequent persistent Betti numbers are always zero if computed on the Reeb graph. Notice also that, for the special case of 2-manifolds embedded in  $\mathbb{R}^3$ , the results on the number of cycles in



**Figure 2.23:** (a,b) Two surfaces studied with respect to the height function  $f$  in the horizontal direction; the values of  $f$  are depicted on the arrows. (c,d) Their corresponding Reeb graphs are not isomorphic, and therefore allow to distinguish the original shapes. (e,f) The 0th and the 1st persistent homology groups of the two surfaces coincide, and therefore are not sufficient to discriminate between these objects.

the Reeb graphs reported by [CMEH<sup>+</sup>03] enabled the extension by [AEHW04] of the notion of persistence to form a pairing between *all* the critical points of a function defined on the manifold.

Skeletons are defined on the basis of the distance function; therefore they are not parametric with respect to the any function  $f$ . Moreover, even if not always linear, they guarantee a dimensionality reduction of the shape domain. However, the exact medial axis guarantees the invertibility and the equivalence of the shape descriptor with respect the original shape. On the contrary, the linearity assumption behind the curve skeleton forces the conciseness and the simplicity of the structure discarding the invertibility of the descriptor.

### 2.5.3 Suitability for applications

From the point of view of applications, Morse and Morse-Smale complexes have proven to be useful tools in analyzing the morphology of terrains. Moreover they naturally provide a shape segmentation, which is suitable both for cutting a surface into a single flattenable piece and for simplifying the model representation through the extraction of a combinatorial base domain. This is fundamental for several geometry processing tasks, such as parameterization, remeshing, surface texturing and deformation. In this context, structural problems like over-segmentation are caused by the presence of noise, and efficiency issues arise because of the very large size of existing data sets; these problems have been faced and solved by using generalization techniques and hierarchical representations. Beside the purpose of visual inspection, in recent works the Morse-Smale complex is computed using the eigenfunctions of the discrete Laplacian operator and used to extract surface quadrangulations, that are stable and intrinsic to the model [DBG<sup>+</sup>06].

Contour trees are mainly exploited in the visualization context. They have become popular in image processing and topography for their properties that allow a real time navigation of the data. In particular, the recent developments on this topic have highlighted their potential for analyzing high-dimensional and time dependent data, like the visualization of the hemoglobin dynamic and the simulation of galaxy formation in the universe, see for example [SB06].

Since Reeb graphs generalize to  $n$ -dimensional manifolds the concepts behind the contour tree, their application domains partially overlap, for instance in scientific visualization. Despite the more general definition, the existing algorithms for Reeb graph extraction mainly work on 2- and 3-manifolds, and only recently on 3D time-dependent data [EHMP04] and  $n$ -dimensional simplicial complexes [PSBM07]. Nevertheless, the definition of Reeb graphs on a domain topologically more “complex” than a scalar field (e.g., with holes, or concavities) emphasizes the compactness of this representation. This fact has stimulated the use of this descriptor in a large number of applications related to surface understanding, simplification, parameterization, segmentation and reconstruction. In addition, the simplicity of the structure (an one-dimensional simplicial complex in every dimension) and the natural link between the properties of the function  $f$  and the shape  $S$  have lead to a massive use of this descriptor for shape comparison and to development of several shape matching and retrieval tools.

Since the beginning, the declared aim of size theory has been the development of a geometrical-topological framework for comparing shapes. Each shape is viewed as a topological space equipped with a real function describing the shape properties relevant to the comparison problem at hand. Measuring dissimilarity in size theory amounts to minimize the change in the functions due to the application of homeomorphisms between topological spaces, with respect to the  $L^\infty$  norm. Size functions are a practical and manageable class of descriptors which allow to provide a lower bound for the dissimilarity measure between shapes. With



this theoretical premise, size functions have been extensively used in the field of Pattern Recognition, mainly for image retrieval and classification in the Computer Vision domain.

The idea of persistent homology was originally introduced to assess the relevance of topological attributes in a growing complex. Persistence furnishes a scale to separate out *topological features*, that is attributes with a long life-time in the growing complex, from *topological noise*. The application of this relevance scale to topological simplification is straightforward, leading for example to methods for reducing noise in sample data. At the same time, considering the life-time of topological attributes also induces a powerful description of the shape under study, that is an invariant reflecting geometrical-topological properties of shapes. This approach to shape comparison has a clear potential which has been demonstrated in a series of examples, although it seems still lacking for an extensive experimentation in comparison with existing techniques. A few years after its introduction, the theory of persistent homology is also becoming one of the basic instruments for solving different application problems, such as protein docking or hole detection in sensor networks.

The MAT is the most popular skeleton representation because it is available both for reconstruction and animation purposes. However, its dependence on small perturbations and noise makes it not very suitable for recognition and matching contexts. Moreover, due to the large computational cost of 3D algorithms (see Table 2.4), it is more used in image contexts than in the polyhedral ones. Unlike other medial axis based algorithms, the curve-skeletons are stable against the noise even though the medial axis may not. Distance field based methods work nicely and efficiently for the tubular objects, see [HF05] for recent results. Since the points with the same distances from the boundary of some shapes may form surface patches, the distance field based methods may face difficulty in extracting curve-skeletons for those shapes. For instance, the approach in [CSYB05] solves the problem by taking into account the surface area. However, in practice, the potential field based method still may fail for the shapes containing thin parts [DS06].

Moreover, depending on the kind of the manipulation task (animation, metamorphosis, growth, etc.), skeletal elements may rotate, stretch, appear or disappear. However, the skeletal elements of the intermediate shapes obtained during the animation evolution remain simply to define, articulate and display and the skeletal hierarchy.

## Related publications

S. Biasotti, D. Attali, J.-D. Boissonnat, H. Edelsbrunner, G. Elber, M. Mortara, G. Santini di Baja, M. Spagnuolo, and M. Tanase. Skeletal structures. In L. De Floriani and M. Spagnuolo, editors, *Shape Analysis and Structuring*, pages 145–183. 2007.

S. Biasotti, L. De Floriani, B. Falcidieno, and L. Papaleo. Morphological representations of scalar fields. In L. De Floriani and M. Spagnuolo, editors, *Shape Analysis and Structuring*,

pages 185–213. 2007.

S. Biasotti, B. Falcidieno, P. Frosini, D. Giorgi, C. Landi, G. Patané, and M. Spagnuolo. 3D shape description and matching based on properties of real functions. In *Eurographics 2007, Tutorial Notes*, pages 949 –998. Eurographics Association, 2007.

S. Biasotti, D. Giorgi, M. Spagnuolo, and B. Falcidieno. Reeb graphs for shape analysis and applications. *Theoretical Computer Science*, 392(1–3):5–22, 2008.

S. Biasotti, B. Falcidieno, L. De Floriani, P. Frosini, D. Giorgi, C. Landi, and M. Spagnuolo. Describing shapes by geometrical-topological properties of real functions. *ACM Computing Surveys*, December 2008. in print.

---

## Chapter 3

# Contribution

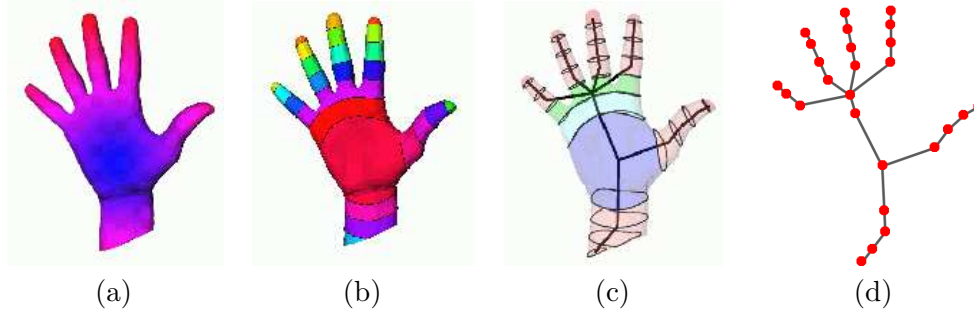
Combining the topological exploration of a shape with quantitative measurement of geometrical properties provided by a real function defined on the shape is crucial for a number of applications, like shape comparison, retrieval and classification.

The added value of approaches rooted on Mathematics in general, and Morse theory in particular, is in the possibility of adopting different functions as shape descriptors according to the properties and invariants that one wishes to analyze. In this sense, it is possible to construct a *general framework* for shape characterization, parameterized with respect to the function  $f$  used, and possibly the space associated with the shape. The function  $f$  plays the role of a *lens* through which we look at the properties of the shape, and different functions provide different insights.

These considerations motivate this work and the definition of a flexible shape description framework that combines mathematical theories with computational constraints. In particular, this general framework may easily be adapted to different contexts by simply modifying or replacing one or more of the ingredients that compose it.

In the general flow that starts from the choice of the shape property (i.e. a function) to analyze and arrives to the definition one or more shape descriptors, in this Chapter we describe two approaches, namely the Shape Graph and the multi-dimensional size descriptor, that code both topological and geometric information. Finally, we introduce also a technique to distinguish among different shape properties and measure how much two functions defined on the same shape differ.

In the reminder of this Chapter, we first introduce the notion of Shape Graph and then discuss two methods to associate geometrical information to the topological graph. Then, the Shape Graph representation is extended to sets of objects and used to extend the size descriptor to three-dimensional data. Moreover, an extension of the computation of the size descriptor to data of any dimension and possibly with respect to an arbitrary number of real



**Figure 3.1:** The distance from the barycenter is highlighted on the hand model (a). The SG nodes correspond to the regions generated from the contours (b). In (c), the SG is superimposed to the model, while the representation in (d) shows all the SG nodes.

functions is proposed. Finally, we introduce a functional that measures the local difference between two functions and, on the basis of this operator, we provide an algorithm to create a function which is almost everywhere orthogonal to a given one.

### 3.1 The Shape Graph

Our *Shape Graph* ( $SG$ ) generalizes the definition of Reeb graph (proposed in Section 2.2.2.1) to a surface on which a finite set of iso-contours (for simplicity contours)  $C(S)$  of a real function  $f$  is defined.

Since contours are supposed to be non degenerate (i.e. points or open lines), they subdivide  $S$  into a set of regions bordered by elements of  $C(S)$ . Then, we define two points  $P, Q \in S$  as equivalent if they belong to the same region or the same contour, see [Bia04a] for details. The quotient space obtained from this relation is a discrete space. In Figure 3.1, an example of the quotient space with respect to the distance from the barycenter is shown for a hand model. Figure 3.1(a) highlights how the function  $f$  varies on the model: blue regions correspond to minima while red ones represent maxima.

This quotient space is coded in a Shape Graph  $G^f = (V, E)$  as follows: first of all, each region  $R$  is represented by a node  $v_R \in V$  located in the barycenter of the region  $R$ ,  $(x_R, y_R, z_R)$ ; then, if two regions share a contour, the nodes corresponding to these regions are connected by an edge  $e_R \in E$ . In general, a node will be linked to as many nodes as the number of components of the border of the associated region. In Figure 3.1(d) it is highlighted how the sequence of points of the quotient space represents the arcs and nodes of the SG.

The underlying slicing mechanisms has to be handled with care: for example, if only a too small number of contours is considered, holes completely contained in the interior of a single region are missed. This problem is related to the slicing frequency, and allows the user to

get rid of little features that are considered irrelevant. Nevertheless, if topological accuracy is required, the problems can be easily overcome with the insertion of an additional number of contours into regions having holes (these regions can be always detected locally in each slice using the Euler Characteristics, see [ABS03] for details). Therefore, even if the contours may be non uniformly distributed on the domain of  $f$ , the SG will correctly represent the topology of the surface.

The computational complexity of the SG extraction is  $O(\max(n \log n, m))$ , where  $n$  is the number of vertices in the original mesh, and  $m$  are the vertices in the mesh after partitioning (see [BGSF06, BGSF08a]). In the worst case  $m$  could become  $O(n^2)$ .

Finally, we notice that, since the description properties of SG derive from the general definition of Reeb graph (see Section 2.2.2.1), when it is necessary that the shape description is invariant under some transformation, like rigid motions or affine transformations, it is sufficient to choose a function which is invariant under those transformations. Therefore, the dependence of the graph on the function provides a flexible shape characterization that can easily be tuned according to the user needs.

### 3.1.1 Coupling the graph representation with geometric attributes

When dealing with applications, the definition of the Shape Graph introduced above, which is mainly topological, may be not sufficient to code the salient shape information. In this case, a set of attributes may be associated both to nodes and edges of the graph. In this Section we will discuss two approaches, the first mainly based on a set of measures of the contours along the graph edges and the second based on the description of the shape region associated to the sub-graph originated by the nodes.

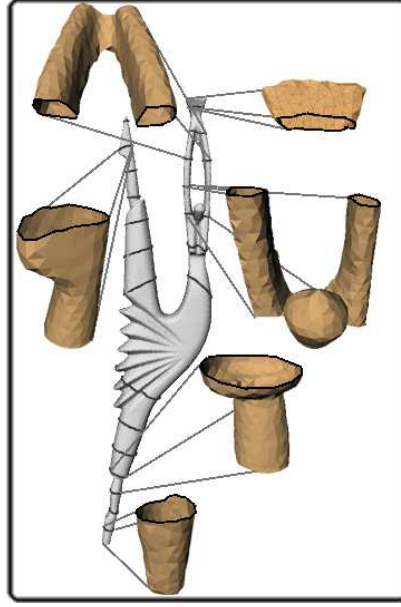
#### 3.1.1.1 Contour-based attributes

We define a set of local geometric attributes to each shape slice that correspond to the SG nodes.

For each node  $v_R \in V(G^f)$  corresponding to a region  $R$ , an attribute of  $R$ ,  $\varphi(v_R)$ , is defined as a property characterizing the region  $R$  or its boundary, as depicted in Figure 3.2. Besides the average of the values that the function  $f$  assumes on the region  $R$ , we store in each node of the SG a vector of measures associated to the geometric attributes of  $R$ . The set of proposed attributes is detailed in the following.

For each node  $v_R \in G^f$  associated to a region  $R$ , a first attribute  $\varphi(v_R)$  can be defined as:

- the *area* of the region  $R$ ,



**Figure 3.2:** Some segments associated to the size graph nodes (graph obtained by segmenting the mesh with the insertion of seven level sets).

that is to say the sum of the areas of the faces in the triangulation which belong to  $R$ . Three other geometric attributes are given by:

- the *minimum* ( $r_{min}$ ), the *maximum* ( $r_{max}$ ) and the *average* ( $r_{av}$ ) *radius* of  $R$ , i.e., the minimum, maximum and average distance of the barycenter  $C = (x_R, y_R, z_R)$  of the triangles in  $R$  from the vertices of the region, see Figure 3.3 (a).

Since the boundary  $B_M(R)$  of a region  $R$  is made of closed contours, the interior of  $R$  is well defined and it is possible to associate to each boundary component a so-called outgoing direction. An outgoing direction is classified as ascending or descending, according to the behavior of the function  $f$  across the corresponding boundary component: the direction is ascending (resp. descending) if the value of  $f$  increases (resp. decreases) walking from the inside to the outside of the region. Let us now denote  $B_M^+(R)$  (resp.  $B_M^-(R)$ ) the set of connected components of  $B_M(R)$  such that the outgoing directions for  $f$  are ascending (resp. descending), see Figure 3.3 (c). For each node  $v_R$  corresponding to a region  $R$ , two attributes can now be defined as:

- the *length* of  $B_M^+(R)$  (resp.  $B_M^-(R)$ ).

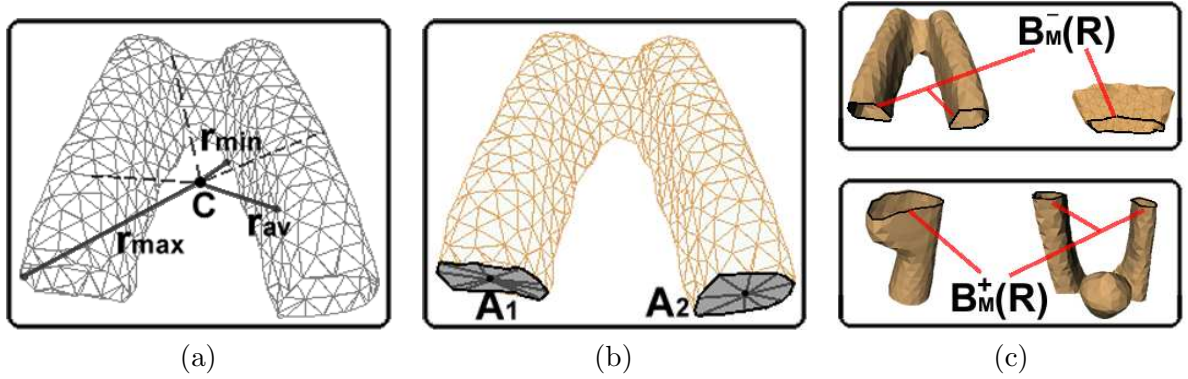
If  $B_M^+(R)$  (resp.  $B_M^-(R)$ ) is made up of multiple components  $\{B_i^+\}$  (resp.  $\{B_i^-\}$ ), then  $\varphi(v_R)$  is the sum of the length of each of the  $B_i^+$ s (resp.  $B_i^-$ s). If all outgoing directions of

$f$  for  $B_M(R)$  are ascending (resp. descending), we label the region  $R$  as a minimum (resp. maximum) for the function  $f$  and assume the length of the border  $B_M^+(R)$  (resp.  $B_M^-(R)$ ) to be zero, that is  $\varphi(v_R) = 0$ .

The next attributes evaluate the lateral area of the pseudo-cone whose basis is the boundary component  $B_i^+$  (resp.  $B_i^-$ ) and whose vertex is the barycenter of  $B_i^+$  (resp.  $B_i^-$ ). Two attributes are indeed defined as

- the sum of the *pseudo-cone areas* computed for each  $B_i^+$  in  $B_M^+(R)$ , (resp.  $B_i^-$  in  $B_M^-(R)$ ), see Figure 3.3 (b).

When all outgoing directions of  $f$  for  $B_M(R)$  are ascending (resp. descending) we set  $\varphi(v_R) = 0$ .



**Figure 3.3:** Details about the geometric meaning of some attributes. (a) Minimum, maximum and average radius of a region. (b)  $A_1$  and  $A_2$  represent two pseudo-conic areas. (c) Length of upper and lower (with respect to a chosen  $f$ ) boundary components.

### 3.1.1.2 Region-based attributes

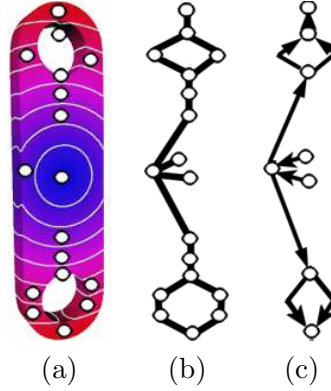
The attributes detailed in Section 3.1.1.1 mainly store a list of border properties of the slices associated to the nodes of the Shape Graph. On the contrary in this Section we deal with attributes related to the shape of the SG regions.

To reduce the number of Shape Graph elements, we have adopted of simplification strategy of the nodes  $V$  and edges  $E$  of  $G^f$ . First, we observe that each edge  $e_R$  of  $G^f$  may be oriented according to the monotonicity of the function  $f$ , which implies that  $G^f$  is directed and acyclic. Furthermore, each node of  $G^f$  identifies a sub-graph which is empty only in case of leaf nodes, that are nodes with out-degree zero.

Then, the SG may be simplified by collapsing all nodes whose number of incoming and outgoing edges is 1, without altering the topological correctness of the coding. After this

merging step, the SG simply consists of nodes representing the regions where the topology of the contours varies and the associated connecting edges, see Figure 3.4(c).

Also in this case, the SG provides a graph representation of the shape that, similarly, to the Reeb graph is able to correctly code the topology of a closed surface [CMEH<sup>+</sup>03] as discussed in [BMSF06].

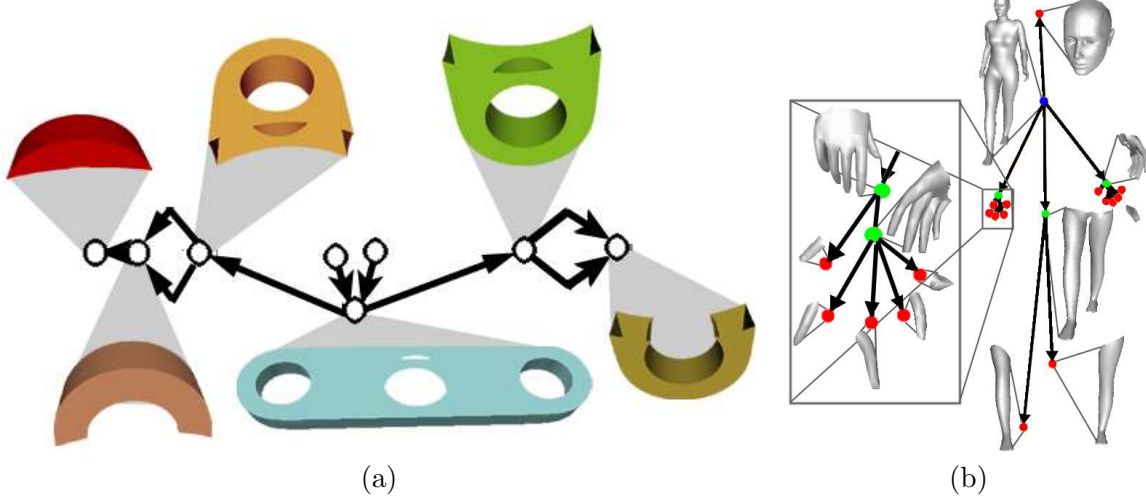


**Figure 3.4:** SG nodes are associated to model regions (a). The SG in (b) is simplified and edges are oriented according to the increasing values of the distance from the barycenter (c).

In this approach we move from a local description of the surface slice to a more general representation of the model sub-parts, based on the assumption that the larger the model portion associated to a node is, the more relevant the node should be. Since the SG is directed, each node is associated to a sub-graph, and this sub-graph defines a sub-part of the shape. The value of  $f$  and a geometric descriptor are associated to each node in the simplified SG. For the nodes whose out-degree is zero (leaf nodes), whose sub-graph is empty, we consider only the slice of the shape that correspond to them. Once sub-parts have been associated to each node, we use the spherical harmonic analysis of the sub-part to describe its geometry. Spherical harmonic analysis has been defined in [KFR03], and this descriptor is rotation and scale invariant and stores the shape distribution of each shape sub-part. Therefore, each node is indexed using a matrix, whose values depend on the spherical harmonic values of the related sub-part. Analogously to the coding of the SG proposed in Section 3.1.1.1, we associate to each node of SG the Cartesian coordinates of the centroid of the corresponding region  $R$  and, mainly for visualization purposes, the node is labelled with type of  $R$  (i.e. maximum, minimum or saddle).

In Figure 3.5, we show the sub-parts associated to the SGs of two models. The SG structure is represented by the graph, and each node is depicted with the sub-part it generates. Figure 3.5(a) corresponds to the SG with respect to the distance from the barycenter of the linkage model in Figure 3.4. Since the holes in the model are not symmetric with respect to the barycenter, the sub-parts associated to the two leaf nodes slightly differ. Figure 3.5(b), depicts the SG associated to a human body model. In both cases, the arrows indicate the



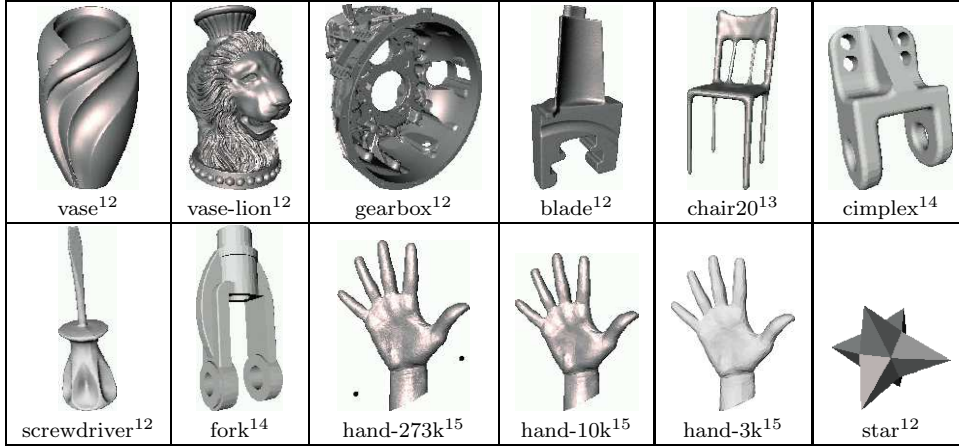


**Figure 3.5:** (a) The SG representation of the model in Figure 3.4 and some of the model sub-parts associated to graph nodes. (b) The structural descriptor of a human body model

edge orientation and shape parts are associated to the SG nodes. Note that there is a node, representing a minimum of  $f$ , whose in-degree is zero: in this case the shape sub-part associated to that node corresponds to the whole model.

Finally, the number of regions crossed during the edge construction is also associated to each edge of the SG, and this value reflects the length of the edge before the simplification step.

As far as the storage space is concerned, the shape descriptor is stored in an ascii VRML file that codes both the graph and the node attributes provided by the spherical harmonics. Table 3.1 provides the time and the storage space needed by our pre-processing step for coding the models depicted in Figure 3.6. Results highlight that the time for coding our structural descriptor depends both on the size of the original model and the shape features with respect to the function chosen. In Table 3.1 the function  $f$  chosen is the distance from the barycenter and graphs are obtained inserting either 16 or 64 contour levels. Comparing the descriptors at two different resolutions, it is reasonable that the number of graph nodes (and the storage size of the descriptor) increases augmenting the number of contour levels used for extracting the SG. In particular, the “hand” models show the stability of the structural descriptor when both the model resolution and the number of contours vary. As pointed out in [GCO06], the partial matching problem requires a large number of comparison operations. Regarding this skill our method is able to evaluate the 176400 comparisons on a database of 420 items (McGill University) in about 7 minutes, which is a rather good time, see the results provided in [ZSM<sup>+</sup>05]. All our experiments were conducted on a PC-3.4Ghz with 2Gb of RAM.



**Figure 3.6:** Models having different size and resolution.

Models	#vertices	#faces	Size	Nodes		Time		Storage size	
				16	64	16	64	16	64
vase	896.338	1.792.672	73MB	6	18	30s	1m31s	79KB	214KB
vase-lion	200.002	400.000	29MB	10	46	11s	45s	123KB	525KB
gearbox	64.147	128.606	8.9MB	32	145	20s	1m18s	375KB	1.7MB
blade	24.998	49992	2.2MB	6	12	2s	5.4s	78KB	145KB
star	14	24	2.1KB	6	6	0.14s	0.36s	79KB	79KB
hand-3k	1.515	3.026	119KB	11	11	0.24s	1s	134KB	134KB
hand-10k	5.023	10.040	438KB	10	13	0.5s	1.7s	123KB	156KB
hand-273k	136.663	273.060	13MB	9	11	10.45s	14.69s	112KB	134KB
chair20	8.456	16.950	689KB	12	19	0.6s	0.9s	145KB	223KB
fork	10.974	21.956	947KB	9	24	1s	6.6s	112KB	279KB
cimplex	8787	17.594	715K	17	23	2.3s	3.8s	190KB	257KB
screwdriver	27.152	54.300	2.1MB	3	9	1s	3.1s	34KB	112KB

**Table 3.1:** Statistics for the models in Figure 3.6.

The structural representation provided by the SG equipped with attributes related to regions may differ from the intuitive notion of shape structure but, since it is related to the mathematical properties of the function  $f$ , it objectively reflects the properties and acts as a filter for matching operations.

The flexibility of the structural descriptor with respect to the choice of the function  $f$  is

<sup>12</sup>These models come from the AIM@SHAPE repository.

<sup>13</sup>This model is a remeshed version of a model in the McGill benchmark.

<sup>14</sup>This model is a remeshed version of a model at Drexel Univ.

<sup>15</sup>These models come from the AIM@SHAPE repository and are provided at three different resolutions.

a characteristic quality that, in this sense, differs from the skeletal decomposition obtained from flow discretization [DGG03] or thinning methods [CSM05, ZSM<sup>+</sup>05] and may become an advantage when the sub-parts to be recognized through the matching have well defined mathematical properties. Another distinctive feature of our descriptor is the use of a descriptor for each node sub-graph (the spherical harmonic transform, [KFR03]) more complex than the usual ones, see Section 4.1.2.

### 3.1.2 Extension to sets of objects

Often in real applications it is important to compare simultaneously sets of elements present in different scenes. For example, this is relevant when a global view of the elements in a scene database is required, maybe for indexing or filtering purposes, before handling a fine recognition analysis. In fact, when massive volumes of data are provided, a fast and high-level analysis of the single scenes could significantly increase the number of computations. The same problem holds when comparing objects made of sets of single parts. In this last case it is necessary to overcome the limitation, typical when using structural descriptors, of comparing objects made of a single connected component. In our approach, we translate the problem of comparing sets of objects that are represented by a SG to the comparison of set of Shape graphs. To extend the Shape Graph representation to set of objects (3D scenes) we have been inspired by the representation proposed in [WHL05].

Wilson et al. [WHL05] showed how graphs can be converted into pattern vectors by using the spectral decomposition of the Laplacian matrix and basis sets of symmetric polynomials. More formally, denoting  $L$  the Laplacian matrix associated to a graph  $G$ ,  $\mathbf{e}_i$  and  $\lambda_i$  respectively the  $i$ -th eigenvector and eigenvalue of  $L$ , the matrix  $L$  may be expressed by the formula:  $L = \Phi \Phi^t$ , where  $\Phi = (\Phi_{i,j})_{i,j=1,\dots,n} = (\sqrt{\lambda_1} \mathbf{e}_1, \dots, \sqrt{\lambda_n} \mathbf{e}_n)$ . Then, a symmetric polynomial in the components of eigenvectors  $\mathbf{e}_i$  is expressed as:

$$\lambda_j = \sum_{i=1}^n \Phi_{ij}^2. \quad (3.1)$$

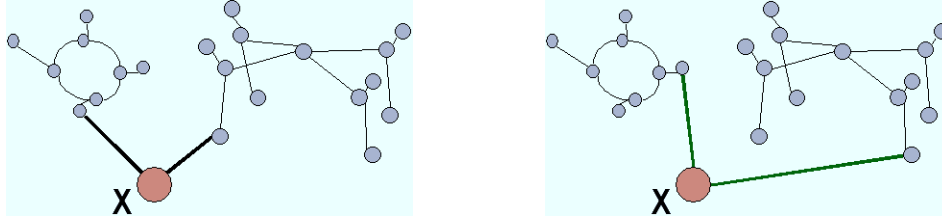
In order to extend this representation to sets of graphs, a virtual node  $X$ , without attributes, is introduced in every set of Shape Graphs (this operation works in analogy to VRML files<sup>17</sup>), and it is joined to one of the vertices of minimum degree for each Shape Graph  $G_i^f$  (see Figure 3.7). Since the Laplacian spectrum is invariant for node label permutations, the choice of the vertex to which  $X$  is joined is not relevant for the extraction of the eigenvalues of  $L$ .

Then, a single and connected graph is obtained, whose Laplacian matrix has the form represented in Equation (3.2), [PBF07b]. If the set is composed by  $m$  SGs  $G_i^f = (V_i, E_i)$ ,

---

<sup>17</sup>A definition of grouping nodes in the VRML format is described at <http://www.agocg.ac.uk/brief/vrml.htm>

---



**Figure 3.7:** Two possible connections between a virtual node  $X$  and a scene made by two graphs. Provided that, for each graph,  $X$  is connected to a node of minimum degree, the representation of the scene graph is independent of the choice of  $X$  (unless of graph isomorphisms).

$|V_i| = n_i$ , with  $i = 1, \dots, m$ , we define the Laplacian matrix of the scene as:

$$L = \begin{pmatrix} m & -1 & \mathbf{0} & \dots & -1 & \mathbf{0} \\ -1 & l_{1,1}^1 + 1 & \dots & l_{1,n_1}^1 & & \\ & \dots & \dots & \dots & & \\ \mathbf{0} & l_{n_1,1}^1 & \dots & l_{n_1,n_1}^1 & & \\ \vdots & & & & \ddots & \\ -1 & & & & l_{1,1}^m + 1 & \dots & l_{1,n_m}^m \\ & & & & \dots & \dots & \dots \\ \mathbf{0} & & & & l_{n_m,1}^m & \dots & l_{n_m,n_m}^m \end{pmatrix}. \quad (3.2)$$

where  $L(G_h^f) = L^h = (l_{i,j}^h)_{i,j=1,\dots,n_h}$  is the Laplacian matrix of the  $h$ -th graph, and the element added to  $l_{1,1}^h \forall h$  stresses the existence of the virtual node joined with a component vertex (we suppose that, after a node label permutation,  $X$  is joined to the first node of the component), whose degree increases of 1.

When dealing with a Shape Graph  $G^f$  having as attributes the values  $\varphi$  described in Section 3.1.1.1, the same procedure described in (3.2) is used to modify the expression of the attributed Laplacian matrix [WHL05] and take into account both node and edge attributes.

In this Section we analyse the relation between the scene graph we have defined and the descriptors of the single scene components. In fact, from successive inequalities, we can find both an upper and a lower bound for the spectrum of  $L$  in (3.2) which depend on the eigenvalues of the Laplacian matrices of the single scene components. Because these graphs are described also by Laplacian eigenvalues, the property we are going to describe guarantees that the scene graph is comparable with those of singular components. Therefore these results justify the addition of a virtual node, instead of separately considering each scene components.

The introduction of  $X$  makes the set of SGs  $G^f$  connected, and it adds a non-zero eigenvalue to its Laplacian matrix  $L$ : this change keeps  $L$  symmetric and diagonally dominant, so positive definite. We have, therefore, analyzed how this operation affects the spectrum of  $L$ .

Let us consider a set with just two graphs  $G_1^f$  and  $G_2^f$ , whose Laplacian matrices are re-

spectively  $L^1$  and  $L^2$ . Denoting  $\lambda_1(L^1), \dots, \lambda_{n_1}(L^1), \lambda_1(L^2), \dots, \lambda_{n_2}(L^2)$  their eigenvalues, let  $\tilde{L}^1 = (\tilde{l}_{i,j}^1)_{i,j=1,\dots,n}$  be such that  $\tilde{l}_{i,j}^1 = \begin{cases} l_{1,1} + 1 & i = j = 1 \\ l_{i,j} & \text{otherwise} \end{cases}$  and similarly  $\tilde{L}^2$ ; it follows that  $\sum_i \lambda_i(\tilde{L}^j) = \sum_i \lambda_i(L^j) + 1$ , because the sum of the eigenvalues is equal to the matrix trace. Moreover, we can factor the Laplacian matrix of the scene  $L$  in (3.2) as  $L = A_1 + A_2$ , such that

$$A_1 = \begin{pmatrix} 1 & -1 & & & \\ -1 & l_{1,1}^1 + 1 & \dots & l_{1,n_1}^1 & \\ & \dots & \dots & \dots & \\ & l_{n_1,1}^1 & \dots & l_{n_1,n_1}^1 & \\ & & & & \mathbf{0}_{n_2} \end{pmatrix}, \quad A_2 = \begin{pmatrix} 1 & & -1 & & \\ & \mathbf{0}_{n_1} & & & \\ -1 & & l_{1,1}^2 + 1 & \dots & l_{1,n_2}^2 \\ & & \dots & \dots & \dots \\ & & l_{n_2,1}^2 & \dots & l_{n_2,n_2}^2 \end{pmatrix};$$

here  $\mathbf{0}_{n_i}$  means a square matrix  $n_i \times n_i$  with all 0 entries.

Now let  $G_3^f$  be the graph  $G_1^f$  with a node  $v$  and an edge  $e$  added ( $E_3 = E_1 + e$ ,  $V_3 = V_1 + v$ ,  $v \notin V_1$ ,  $e = (v, w)$ ,  $w \in V_1$ ), from the interlace theorem [Moh91] it follows:

$$0 = \lambda_1(L^3) = \lambda_1(L^1) \leq \lambda_2(L^3) \leq \lambda_2(L^1) \leq \dots \leq \lambda_{n_1}(L^3) \leq \lambda_{n_1}(L^1) \leq \lambda_{n_1+1}(L^3),$$

where  $L^3$  denotes the Laplacian matrix of  $G_3^f$ . As a consequence, the non-zero eigenvalues of  $A_1$  interlace those of  $\tilde{L}^1$ , and the same is valid for  $A_2$  and  $\tilde{L}^2$ .

Denoting  $\lambda_N(L)$ ,  $N = n_1 + n_2 + 1$ , the maximum eigenvalue of  $L$ , Weyl's inequality (see for example [Fra93]) implies the following relation:  $\lambda_N(A_i) \leq \lambda_N(L) \leq \lambda_N(A_1) + \lambda_N(A_2)$ , for  $i = 1, 2$ , since both  $A_1$  and  $A_2$  are positive semi-definite. Therefore it follows:

$$\max\{\lambda_N(A_1), \lambda_N(A_2)\} \leq \lambda_N(L) \leq \lambda_N(A_1) + \lambda_N(A_2). \quad (3.3)$$

In addition we observe that, even if the eigenvalues of  $L$  are always non-zero ( $L$  positive definite), the eigenvalues of the matrices  $A_{i=1,2}$  factorizing  $L$  verify  $\lambda_1(A_i) = \dots = \lambda_{N-1-n_i}(A_i) = 0$ ,  $i = 1, 2$ . Moreover, Weyl's inequality gives a meaningful result when the index  $i$  is  $\max\{n_1 + 1, n_2 + 1\}$ ; if  $n_1 > n_2$  then

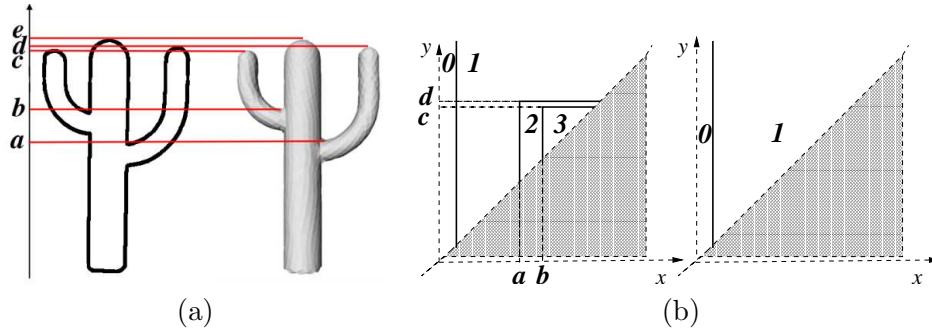
$$\lambda_{n_1+1}(L) \geq \max\{\lambda_{n_1+1}(A_1), \lambda_{n_1+1}(A_2)\}.$$

Analogous results hold for scenes with  $m$  elements, that is  $L = A_1 + \dots + A_m$ , and for attributed graphs. The relation 3.3, in particular, implies that the eigenvalues of the Laplacian matrix of a scene graph  $G^f$  are bounded by the eigenvalues of their components and their sum.

## 3.2 Higher-dimensional size functions

The second shape descriptor we are considering in this thesis are the size functions; in particular we have tackled the problem of extending their computation to 3D shapes.

While in principle the space dimension can be arbitrarily large, in practice the performance of size functions may be weaker if they are directly applied to the given space. As an example, let us consider the two different representations of the cactus shape shown in Figure 3.8 (a): a planar 1D contour sketching the profile (left) and a 2D surface (right). If we describe the 1D contour by means of the height function, we obtain a size function which is sufficiently informative about the shape (see Figure 3.8 (b-left)). The same description for the 2D surface produces a trivial size function (see Figure 3.8 (b-right)), since the number of connected components for the lower level set  $\{P : \varphi(P) \leq y\}$  is always 1, for any value of  $y$ ,  $y \geq \min_P \varphi(P)$ . The solution comes from the identification of the “right” set of shape properties, i.e. the “right” size pairs, that are suitable for the problem at hand.



**Figure 3.8:** (a) Two different representations of a cactus shape and (b) their corresponding size functions.

### 3.2.1 Size functions of 3D objects from topological graphs

To overcome the limits of a straightforward application of size functions to 3D shapes, our idea is to associate with a 3D object a size graph  $(G^f, \varphi)$ , where  $G^f$  is a shape graph representing the 3D object,  $f$  is a real function driving the Shape Graph extraction and called the mapping function, and  $\varphi$  is a measuring function labelling each node of the graph with local geometrical properties of the original model [BGSF06, BGSF08a]. The replacement of a 3D shape with this auxiliary space  $(G^f, \varphi)$ , which couples the structural information computed by the function  $f$  with the different information provided by the measuring function  $\varphi$ , produces non-trivial size functions. The use of  $(G^f, \varphi)$  reduces the dimensionality of the problem, meanwhile storing a rich set of information about the original object.

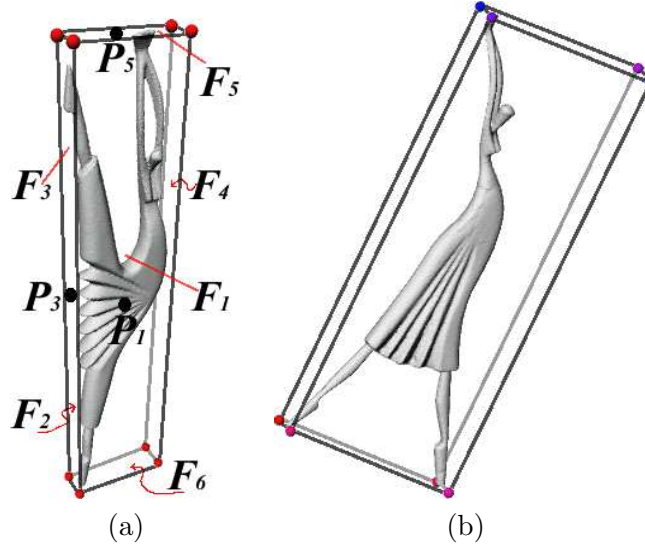
Two ingredients contribute to the definition of size graphs  $(G^f, \varphi)$ : the procedure to build the Shape Graph  $G^f$  starting from the level sets of an appropriate mapping function  $f$  and the definition of a set of measuring functions  $\varphi$  to associate the most appropriate and effective geometric descriptors with  $G^f$ .

In our approach, the Shape Graph is built using the technique described in Section 3.1.1.1. Then, as measuring functions we have adopted the  $\varphi$  attributes of the contour-based SG representation described in Section 3.1.1.1. Besides the geometric attributes of the model, a further measuring function is given by the location in space of the object.

As previously discussed, each node  $v_R$  of the SG is located in the barycenter  $(x_R, y_R, z_R)$  of the corresponding region  $R$ . In addition, for each model, we compute its minimal bounding box following the algorithm in [BH01], which provides a bounding box oriented towards the principal direction of the object (as shown in Figure 3.9). The faces  $F_i$ ,  $i = 1, \dots, 6$ , of the bounding box are ordered according to their increasing areas, thus obtaining the ordered set of pairs  $\{(F_1, F_2), (F_3, F_4), (F_5, F_6)\}$ , where each pair corresponds to a pair of opposite faces, having the same area. Then, the center  $P_i$  is evaluated for each face  $F_i$ ,  $i = 1, \dots, 6$ , see Figure 3.9 (b), and a set of measuring functions  $\{\varphi_{P_i}\}$  can be constructed by computing the Euclidean distance  $\varphi_{P_i}(v_R) = |v_R - P_i|_E$ . The invariance with respect to axial symmetry is obtained replacing the set  $\{\varphi_{P_i}\}$  by the set of measuring functions  $\{\varphi_{P_i, P_{i+1}}\}$ , with  $i = 1, 3, 5$ , and such that:

$$\varphi_{P_i, P_{i+1}}(v_R) = \min\{|v_R - P_i|_E, |v_R - P_{i+1}|_E\},$$

where the minimum distance is considered between the node  $v_R$  and the centers of two opposite faces.



**Figure 3.9:** (a,b) Minimal bounding boxes of two models.

After a size graph  $(G^f, \varphi)$  has been obtained, with  $G^f$  the shape graph and  $\varphi$  the measuring function labeling its nodes, the definition of the size function of the size graph follows the classical one. Denoting by  $G_y^f$  the subgraph of  $G^f$  obtained by erasing all vertices of  $G^f$  at which the measuring function  $\varphi$  takes a value strictly greater than  $y$ , and all edges that connect those vertices to other vertices, the size function of the size graph  $(G^f, \varphi)$  is defined

by setting  $\ell_{(G^f, \varphi)}(x, y)$  equal to the number of connected components of  $G_y^f$ , containing at least a vertex of  $G_x^f$ .

In order to compute size functions, we follow the algorithm based on the so-called  $\Delta^*$ -*reduction* technique introduced in [d'A00] and described in Section 2.3.1.2. The process works in a finite number of operations, and its output is a graph that has the simple structure of a tree.  $\Delta^*$ -reduction permits direct computation of the cornerpoints and cornerlines, that completely describe the size functions (see Section 2.3.1.1). To achieve this, one has to orient the reduced graph obtained through the process of  $\Delta^*$ -reduction by orienting each edge from the vertex with higher value to the other one. The resulting configuration is an arborescence, i.e. an oriented tree in which no two edges are directed to the same vertex. Finally, the cornerpoints and cornerlines are computed from this arborescence by applying the following recursive procedure: (i) choose the highest leaf  $v$  and erase it together with the corresponding edge  $uv$ ; (ii) put a cornerpoint at  $(\varphi(v), \varphi(w))$ ; (iii) if just one vertex  $u$  is left, then draw the cornerline  $x = \varphi(u)$  and stop, otherwise repeat from (i).

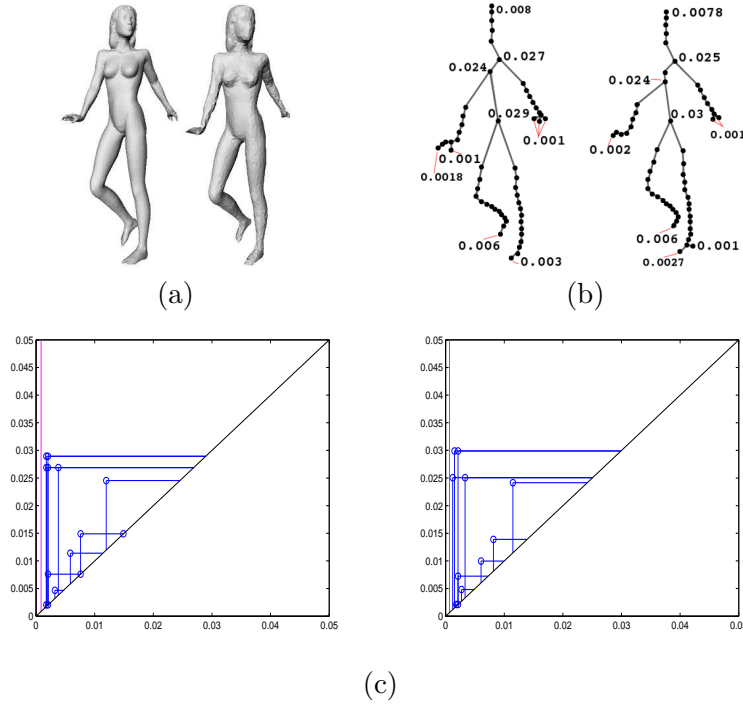
A possible implementation for this algorithm is based on the union-find structure [TvL84], so that the computational complexity for the whole procedure is  $O(n \log n + m \cdot \alpha(2m + n, n))$ , where  $n$  and  $m$  are the number of vertices and edges in the size graph, respectively, and  $\alpha$  is the inverse of the Ackermann function [Ack28].

Figure 3.10 (a) shows a model and its simplified version (having 10% of the vertices of the original model). The difference in the models neither significantly alters the attribute Shape Graph (Figure 3.10 (b)) nor affect the corresponding size function (Figure 3.10 (c)), and the resulting matching distance is consistent. In practice, the method reveals to be robust from different perspectives:

1. *size graphs*: the attributed Shape Graph representation is stable under small variations of the mapping function  $f$  (Figure 3.10 (b)). In fact, it is possible to adopt a refinement of the slicing strategy, that guarantees that all features having size (in terms of the variation of the mapping function  $f$ ) greater than a given threshold are detected, discarding the features whose size is smaller, see discussions in [BFS04];
2. *size functions*: small changes of the size graph do not significantly alter the size functions (that can almost be superimposed in Figure 3.10 (c));
3. *matching*: the stability of matching distance between size functions under small perturbations of the data is confirmed by the example shown, where the matching distance between the size functions computed for the two models is very small.

The results of a second check for robustness are shown in Table 3.2, where the distances between six different objects in our database (four humans and two manufactured models) are reported. In particular, two of the human models are simplified versions of the other

















**Figure 3.10:** Changes due to a simplification of a model do not significantly alter the Shape Graph and its size function. (a) A model and its simplified version; (b) their size graphs with some attributes highlighted, using the integral geodesic distance  $f^3$  and the measuring function area; (c) the corresponding size functions, which show very small variations.

two ones, while the manufactured models differ for some small features. As expected, the comparison framework satisfies the identity property, guaranteeing that a model has a null distance from itself. In addition, the distance between a model and its slightly modified version is smaller than the distance between two different objects in the same class (see for example the woman and the man models) and significantly smaller than the distance between objects belonging to different classes (e.g. a human and a manufactured model). We remark here that these distances are used to rank the results of database queries and aggregate the models that share analogous properties, although they cannot be interpreted as “absolute” values, due to their dependence on the values of the measuring function.

In conclusion, the proposed Shape Graph presents many desirable properties. Indeed it is:

1. quick to compute: the computation of 120 size functions for the 120 models in the database requires 1.53 second on a 1.73GHz laptop PC-M; the off-line step of computing the size graphs requires 1 minute and 12 seconds;
2. concise to store, requiring less than 1k storage per model (see details in Table 3.3);

						
	0.000	0.003	0.021	0.019	0.286	0.241
	0.003	0.000	0.021	0.020	0.286	0.241
	0.021	0.021	0.000	0.010	0.286	0.241
	0.019	0.020	0.010	0.000	0.286	0.241
	0.286	0.286	0.286	0.286	0.000	0.079
	0.241	0.241	0.241	0.241	0.079	0.000

**Table 3.2:** Values for the matching distances between six different models in our database. The size graphs have been obtained using the integral geodesic distance  $f^3$  and the region area.

3. easy and quick to compare: evaluating  $120 \times 120$  matching distances between size functions requires 8.55 seconds;
4. invariant under similarity transformations: imposing the desired invariance simply means requiring the invariance for the mapping and measuring functions, without any change in the mathematical model;
5. robust against noise and small extra features as shown in Figure 3.10 and Figure 3.2.
6. able to discriminate among shapes at many scales, conveying information about global and local properties of the shape, as shown by the experimental results.

The added value of our approach relies in the fact that we provide a modular framework based on the idea of describing shapes by geometrical-topological properties of real functions. A suite of descriptors is thus available, which can be fit to the problem at hand. We also believe that, far from being exhaustive, the set of proposed shape properties (that is, the set of chosen real functions) produces a promising set of descriptors, which helps the user to tune the retrieval system to be on his/her wavelength.

As for the limitations of our approach, currently the method can process only manifold meshes; therefore, polygon soups and point cloud models are not admissible. However,

Models	#vertices	#faces	Size	#N	#CP	SF
vase [aim]	896.338	1.792.672	73MB	45	9	1k
Happy Buddha [sta]	543.652	1.087.716	45MB	27	12	1k
armadillo [aim]	165.951	331.898	15MB	41	12	1k
hand [aim]	136.663	273.060	13MB	56	13	1k
Bunny [sta]	32.872	65.740	3.1MB	18	4	1k
dancer [aim]	24.998	49.996	2.2MB	22	6	1k
dancer2 [aim]	26.358	52.716	2.6MB	20	6	1k

**Table 3.3:** Statistics for some models, relating the dimension of the original model with the number of nodes ( $\#N$ ) of the size graph  $G^f$ , the number of cornerpoints ( $\#CP$ ) of the corresponding size function  $SF$  and the storage size of the final descriptor.

methods able to analyze these classes of representations must necessarily rest on rough approximations of the shape that could discard local shape characteristics. Indeed, the power of this approach is to exploit a number of topological and geometric insights and to keep them during the comparison process, without depending on the uniqueness and the connectivity of the mesh elements. In this sense our framework furnishes an abstract shape description, which is not in contrast with the existing methods but may be fruitfully coupled with them.

### 3.2.2 Multidimensional size functions

More in general, an important problem is given by the possibility of working in a  $k$ -dimensional setting, that is, using multivariate functions with values in  $\mathbb{R}^k$ . Multi-dimensional functions and, consequently, multi-dimensional size theory and natural pseudo-distance were introduced in [FM99]. However, a direct approach to the multi-dimensional case implies working in subsets of  $\mathbb{R}^k \times \mathbb{R}^k$ , and it is unclear how to combinatorially represent multi-dimensional size functions.

A solution is proposed in [CBG07, BCF<sup>+</sup>07], where it is proven that the computation and comparison of multi-dimensional size functions can be reduced to the usual (one-dimensional) case by a suitable change of variables. The idea is that the domain of  $k$ -dimensional size functions can be suitably partitioned into half-planes, such that the restriction of a  $k$ -dimensional size function to each half-plane is a classical one-dimensional size function. This implies that, on each half-plane of the domain partition, the size functions can be represented by cornerpoints and cornerlines. A multi-dimensional matching distance can also be defined, based on the one-dimensional matching distance on each half-plane, which is stable for small changes in the measuring functions.

Since every one-dimensional size function (1SF) may be seen as a linear combination of

triangles (formal series of function characteristics), the comparison of two 1SF's is simple and computationally efficient [dFL05] (see also Section 2.3.1.2). Unfortunately, the same properties does not hold for kSF's. Moreover, a direct approach to the multidimensional case implies working in subsets of  $\mathbb{R}^k \times \mathbb{R}^k$ : In this case, the absence of a compact representation for  $k$ -dimensional size functions involves great efforts from a computational point of view. All these problems have been overcome from the theoretical point of view in [BCF<sup>+</sup>07] by means of a suitable change of variables that allows us to reduce  $k$ -dimensional size functions to the 1-dimensional case. Indeed, it has been demonstrated that there exists a parameterized family of half-planes in  $\mathbb{R}^k \times \mathbb{R}^k$  such that the restriction of  $\ell_{(D,\vec{\varphi})}$  to each of these planes can be seen as a particular 1-dimensional size function.

Let  $(D, \vec{\varphi})$  be a size pair, with  $\vec{\varphi} = (\varphi_1, \dots, \varphi_k) : D \rightarrow \mathbb{R}^k$ . We shall call *admissible pair* any pair  $(\vec{l}, \vec{b}) \in \mathbb{R}^k \times \mathbb{R}^k$  with  $\vec{l}$  unit vector such that  $l_i > 0$ ,  $i = 1, \dots, k$ , and  $\sum_{i=1}^k b_i = 0$ . The set of all admissible pairs will be denoted by  $\text{Adm}_k$ . In this setting, consider the foliation of the open set  $\Delta^+ = \{(\vec{x}, \vec{y}) \in \mathbb{R}^k \times \mathbb{R}^k : \vec{x} \prec \vec{y}\}$  given by the parameterized family of half-planes  $\{\pi_{(\vec{l}, \vec{b})}\}_{(\vec{l}, \vec{b}) \in \text{Adm}_k}$  defined by the parametric equations:

$$\begin{cases} \vec{x} = s\vec{l} + \vec{b} \\ \vec{y} = t\vec{l} + \vec{b} \end{cases}$$

with  $s, t \in \mathbb{R}$ ,  $s < t$ . The restriction on the choice of  $\vec{l}$  and  $\vec{b}$  guarantees a unique parametric representation for each half-plane  $\pi_{(\vec{l}, \vec{b})}$ . Under these assumptions, in [BCF<sup>+</sup>07] the following result has been proved:

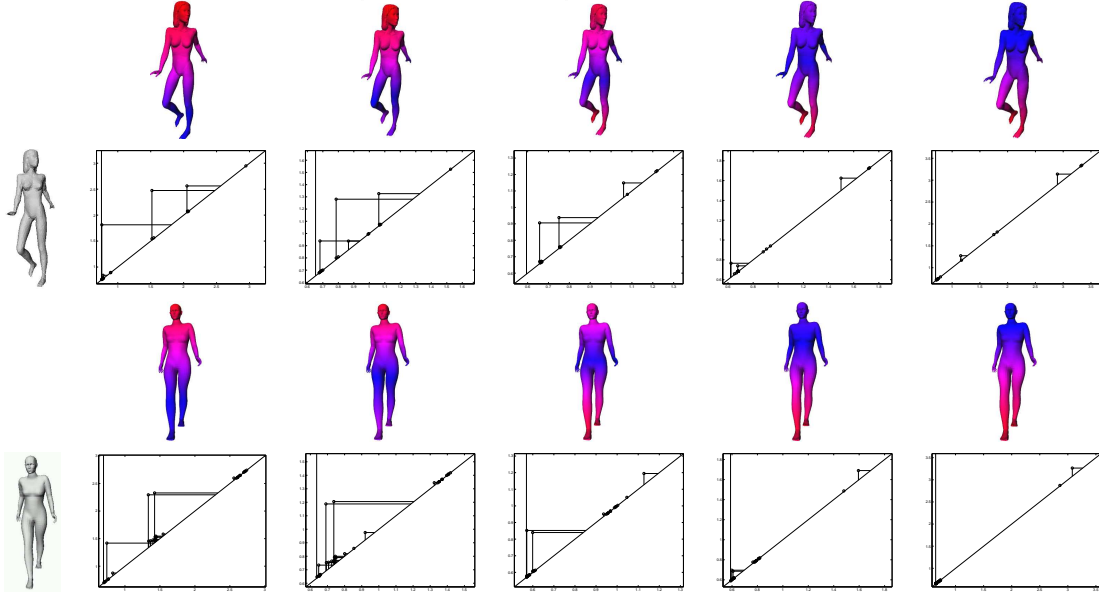
**Theorem 3.2.1** *Let  $(\vec{l}, \vec{b})$  be an admissible pair, and  $F_{(\vec{l}, \vec{b})}^{\vec{\varphi}} : D \rightarrow \mathbb{R}$  be defined by setting*

$$F_{(\vec{l}, \vec{b})}^{\vec{\varphi}}(P) = \max_{i=1, \dots, k} \left\{ \frac{\varphi_i(P) - b_i}{l_i} \right\} .$$

*Then, for every  $(\vec{x}, \vec{y}) = (s\vec{l} + \vec{b}, t\vec{l} + \vec{b}) \in \pi_{(\vec{l}, \vec{b})}$  the following equality holds:  $\ell_{(D, \vec{\varphi})}(\vec{x}, \vec{y}) = \ell_{(D, F_{(\vec{l}, \vec{b})}^{\vec{\varphi}})}(s, t)$ .*

In other words, Theorem 3.2.1 states that a foliation of  $\Delta^+$  in half-planes can be given, such that the restriction of a  $k$ -dimensional size function to these half-planes turns out to be a classical size function in two scalar variables. This result implies that each size function, with respect to a  $k$ -dimensional measuring function, can be completely and compactly described by a parameterized family of discrete 1SF's.

An example of the computation of multi-dimensional size functions is shown in Figure 3.11. Two surface models are analyzed using a two-dimensional measuring function  $\vec{f} = (f_1, f_2)$ , and the domain of the two-dimensional size function is partitioned into half-planes. Details on the definition of  $\vec{f}$  and the partition can be found in [BCF<sup>+</sup>07]. In Figure 3.11, we show









**Figure 3.11:** Example of the computation of a two-dimensional size function on two surface models, on five half-planes of the domain partition defined in [BCF<sup>+</sup>07].

the behavior of the corresponding two-dimensional size function on five half-planes of the partition, depicted from left to right. On each half-plane, the two-dimensional size function coincides with a classical one-dimensional size function. The underlying one-dimensional measuring functions, derived from the components of  $\vec{f}$ , are depicted on top of the corresponding size functions (red corresponds to high values of the measuring function, blue corresponds to low values). Then, the two-dimensional matching distance can be computed, based on the classical one-dimensional matching distances between the size functions on each of the half-planes.

The results proposed in Table 3.4 describe the dimension of some digital models, the average time taken by our algorithm to extract the 1-dimensional size function on a half-plane of the foliation, the total time required to compute the size function on 9 half-planes, and the average and the total number of cornerpoints of the size function on 9 half-planes. These results are obtained using a AMD Athlon 3500, with 2 GB RAM.

### 3.3 Comparison of real functions

In Section 1.3 we have seen that a large number of real functions is available for shape description purposes and, in general, there is not the best choice but it depends on the user's needs. What is interesting is to analyze if two functions, even if not numerically identical, have the same global properties and if it is possible, starting from a real function  $f$ , to create

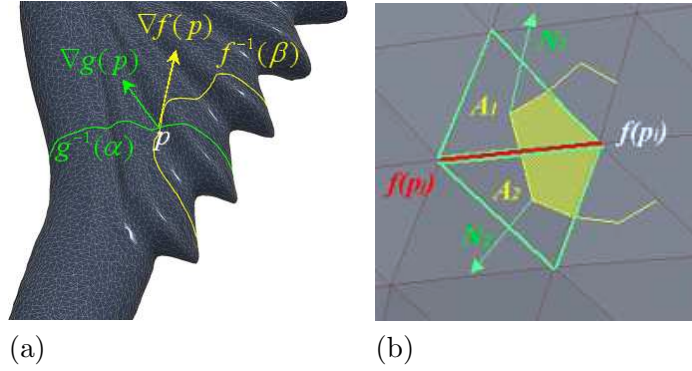
Model	$ V $	$ E $	Avg. time	Total time	Avg. $ C $	Total $ C $
	19538	187005	0.035s	0.315s	180	1623
	18779	193181	0.041s	0.369	84	756
	11504	97757	0.015s	0.135	16	142
	29061	289461	0.056s	0.504	15	133
	114277	1233063	0.212s	1.908	28	254
	5472	45689	0.006	0.054	14	122

**Table 3.4:** Time requirements for the computation of the size function of some 3D images of different dimensions.  $|V|$  and  $|E|$  represent the number of vertices and edges of the size graphs of the models. Avg. time is the average time required to compute the size function on a single half-plane of the foliation, while Total time refers to the computation of the size functions on 9 half-planes. Analogously, Avg.  $|C|$  is the average number of cornerpoints of the size function on a single half-plane of the foliation, and Total  $|C|$  is the sum of the number of cornerpoints of the size functions on 9 half-planes.

a new one  $g$  that measures shape features completely different from those measured by  $f$ .

In our approach, we judge that two functions defined on the same shape  $S$  are *similar* if they have a similar behavior on the same regions of  $S$  and we estimate this similarity by studying the differences of the level sets associated to both  $f$  and  $g$ . Our assumption is that a big difference of the level sets of the measuring functions indicates a significant difference of their behavior on  $S$  [BPSF07].

More precisely, to evaluate the *similarity* of the functions  $(S, f)$  and  $(S, g)$ , we define a new functional  $\mathcal{I}(f, g) : S \rightarrow \mathbb{R}$  that measures the *angle variation* of their gradient fields, Section 3.3.1. As original contributions with respect to the previous work, we provide a direct relation between the critical points of  $f$ ,  $g$ , and  $\mathcal{I}(f, g)$ . Then, we generalize this problem to an arbitrary number of functions defined on  $S$ . Furthermore, we extend the functional  $\mathcal{I}(f, g)$  to discrete domain; given a scalar field  $\Gamma = (D, f)$ , we also use the similarity functional to



**Figure 3.12:** (a) Iso-contours of two functions  $f$  and  $g$  that intersect at  $p$ . (b) Discretization of the gradient field of  $f$  at  $p_i$  with respect to its 1-star.

calculate a new function  $g : D \rightarrow \mathbb{R}$  that is “orthogonal” to  $(D, f)$  (i.e., the most dissimilar) with respect to  $\mathcal{I}$  and we provide an efficient algorithm for its computation. In this way, we also provide a locally orthogonal coordinate system on the surface, a property vital for various applications, like a consistent computation of geodesic distances on the surface.

### 3.3.1 Continuous case

Let  $S \subset \mathbb{R}^3$  be a 2-manifold equipped with and two real functions  $f, g : S \rightarrow \mathbb{R}$  of class  $C^k$ ,  $k \geq 1$ . The *gradient* of  $f$  is defined as  $\nabla f := (\partial_{x_1} f, \partial_{x_2} f, \partial_{x_3} f)$  and its magnitude gives the slope of  $f$  when moving along the normal vector to  $S$ .

We compare the functions  $f$  and  $g$  by studying the bilinear functional (see Figure 3.12(a))

$$\mathcal{I}(f, g) := \langle \nabla f, \nabla g \rangle .$$

From the previous definition, it follows that  $\mathcal{I}(f, g)$  is null at those points of  $D$  where  $\nabla f$  is orthogonal to  $\nabla g$  and at the critical points of  $f$  or  $g$ . We now characterize  $\mathcal{I}$  by analyzing its critical points and establishing their relations with those ones of  $f$  and  $g$ . In matrix form, the gradient of  $\mathcal{I}(f, g)$  may be expressed as:

$$\nabla \mathcal{I} = \mathbf{H}(f) \nabla g + \mathbf{H}(g) \nabla f, \quad (3.4)$$

where  $\mathbf{H}$  represents the Hessian matrix  $((\partial_{x_i x_j} \cdots)_{ij})$  of a function.

From (3.4), it follows that:

- $p \in S$  is critical for  $\mathcal{I}$  if and only if  $\mathbf{H}(f) \nabla g + \mathbf{H}(g) \nabla f = 0$  at  $p$ . Therefore, there might exist critical points of  $\mathcal{I}$  that are not critical of  $f$  and  $g$ ;
- if  $p \in S$  is a critical point of  $f$  and  $g$ , then  $p$  is critical for  $\mathcal{I}$ ;

- if  $p \in S$  is a critical point of  $f$  and  $\mathcal{I}$  but not of  $g$ , then  $p$  is a degenerate critical point of  $f$ .

We define as *averaged dissimilarity measure* between  $f$  and  $g$  on  $S$ , the real number:

$$\mathcal{I}^*(f, g) := \frac{1}{\text{area}(S)} \int_S \underbrace{\langle \nabla f, \nabla g \rangle}_{\mathcal{I}(f, g)} d\mathbf{p}. \quad (3.5)$$

Since  $\text{div}(f\nabla g) = \langle \nabla f, \nabla g \rangle + f\Delta g$ , we get that if  $g$  is harmonic (i.e.,  $\Delta g = 0$ ) then  $\text{div}(f\nabla g) = \langle \nabla f, \nabla g \rangle$ ; by integrating the previous identity on  $S$ , we have

$$\mathcal{I}^*(f, g) = \frac{1}{\text{area}(S)} \int_S \text{div}(f\nabla g) d\mathbf{p}.$$

Finally, we introduce the *normalized dissimilarity measure* as

$$\bar{\mathcal{I}}(f, g) := \frac{1}{\text{area}(S)} \int_S \left\langle \frac{\nabla f}{\|\nabla f\|}, \frac{\nabla g}{\|\nabla g\|} \right\rangle d\mathbf{p},$$

and the *similarity measure* is defined as  $1 - \bar{\mathcal{I}}(f, g)$ .

### 3.3.2 Discrete case

This Section presents the discrete counterpart of the concepts and descriptors defined in Section 3.3.1.

To define the functional (3.5) on a triangle mesh  $D$ , we approximate the gradient of  $f$  at  $\mathbf{p}_i$  as [Hir03]

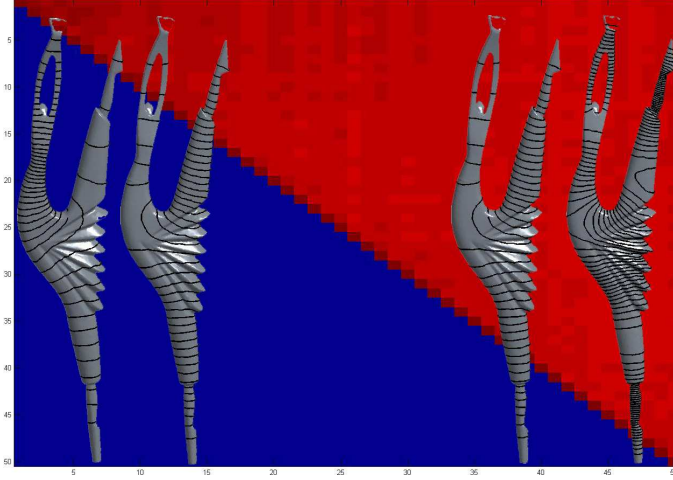
$$\nabla f(\mathbf{p}_i) := \sum_{j \in N(i)} [f(\mathbf{p}_j) - f(\mathbf{p}_i)] \mathbf{w}_j, \quad \mathbf{w}_j := \frac{1}{3} \mathbf{N}_1 + \frac{1}{3} \mathbf{N}_2 \quad (3.6)$$

where  $\mathbf{N}_1, \mathbf{N}_2$  (resp.,  $A_1, A_2$ ) are the normal vectors (resp., the area of the Voronoi regions) of the two triangles which share the edge  $(i, j)$ , and  $N(i) := \{j : (i, j) \text{ is an edge}\}$  is the 1-star of the vertex  $i$  (see Figure 3.12(b)). We explicitly note that the vectors  $\mathbf{w}_j$ ,  $j \in N(i)$ , do not depend on  $f$  and (3.6) fulfills the main properties that apply to the gradient in the continuous case, that is, linearity and nullity (i.e.,  $f = \text{const}$  implies  $\nabla f = 0$ ).

We now generalize the comparison to an arbitrary number of functions on  $D$ . Given  $n$  mapping functions  $f_1, \dots, f_n$  on  $D$ , we introduce the symmetric  $n \times n$  matrix  $\mathbf{A} := (a_{ij})$ , where  $a_{ij} := |\bar{\mathcal{I}}(f_i, f_j)|$ , and we define as *correlation factor* of  $\{f_i, i = 1, \dots, n\}$  the standard deviation of the set  $\{a_{ij} : i = 1, \dots, n, j \geq i\}$ , that is,

$$\sigma(f_1, \dots, f_n) := \sqrt{\frac{\sum_{i=1}^n \sum_{j \geq i} [a_{ij} - \bar{a}]^2}{n}}, \quad \text{with } \bar{a} := \frac{1}{n(n+1)} \sum_{i=1}^n \sum_{j \geq i} a_{ij}.$$





**Figure 3.13:** Color image of the matrix  $\mathbf{A}$  related to the first 50 eigenfunctions of  $D$ ; on the left (resp., right) pairs of Laplacian eigenfunctions  $(f_1, f_2)$  (resp.,  $(f_2, f_6)$ ) with the lowest (resp., highest) dissimilarity measure  $\overline{\mathcal{I}}$ .

Since the vectors  $\mathbf{w}_j$ ,  $j = 1, \dots, n$ , do not depend on the input function, the entries of  $\mathbf{A}$  can be efficiently calculated by storing the coefficients  $\{ \langle \mathbf{w}_i, \mathbf{w}_j \rangle : i = 1, \dots, n, j \in N(i), j > i \}$  and using matrix multiplications. We note that through the matrix  $\mathbf{A}$  (see Figure 3.13) we can identify the function  $f_j$  which mostly differs from  $f_i$  (i.e.,  $j = \operatorname{argmin}_k \{a_{ik}\}$ ), as well as the pairs of functions with a similar (resp., dissimilar) behavior, i.e.  $(f_i, f_j)$  such that  $a_{i,j} = \max_{p < q} \{a_{pq}\}$  (resp.,  $a_{i,j} = \min_{p < q} \{a_{pq}\}$ ).

### 3.3.2.1 Almost-everywhere orthogonal function

Given an arbitrary function  $f$ , we consider the following problem: *find  $g : D \rightarrow \mathbb{R}$  such that  $\nabla g$  is orthogonal to  $\nabla f$  on  $D$*  and we prove that it always has the trivial solution  $g = \text{const}$ . By imposing the conditions:

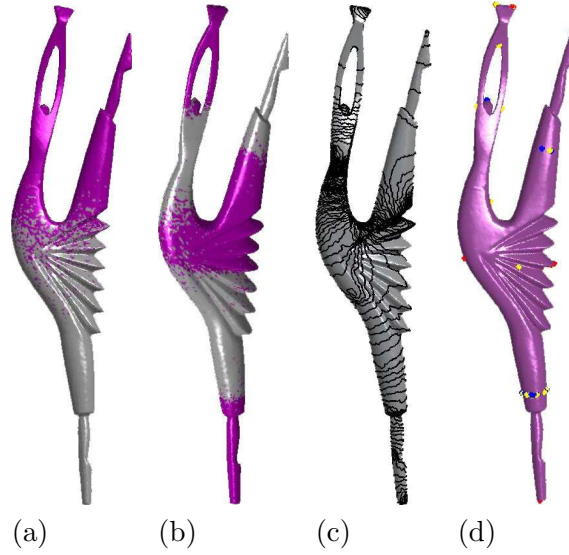
$$\langle \nabla f(\mathbf{p}_i), \nabla g(\mathbf{p}_i) \rangle = 0, \quad i = 1, \dots, n, \quad (3.7)$$

where  $\{\nabla f(\mathbf{p}_i)\}_{i=1, \dots, n}$  are constant vectors and  $\{g(\mathbf{p}_i)\}_{i=1, \dots, n}$  are the unknowns, we get  $n$  linear equations

$$\sum_{j \in N(i)} \langle \nabla f(\mathbf{p}_i), \mathbf{w}_j \rangle g(\mathbf{p}_j) - \sum_{j \in N(i)} \langle \nabla f(\mathbf{p}_i), \mathbf{w}_j \rangle g(\mathbf{p}_i) = 0$$

with  $i = 1, \dots, n$ . These relations can be written in matrix form as:

$$\mathbf{A} \mathbf{g} = \mathbf{0} \quad (3.8)$$



**Figure 3.14:** (a-b) Variation of  $\bar{T}$  on  $D$  for the pairs of functions in Figure 3.13; moving from white to pink the dissimilarity of the functions increases. (c) Iso-contours of the function orthogonal to  $f_1$ ; (d) critical points of  $f_1$  and visualization of  $\bar{T}$ .

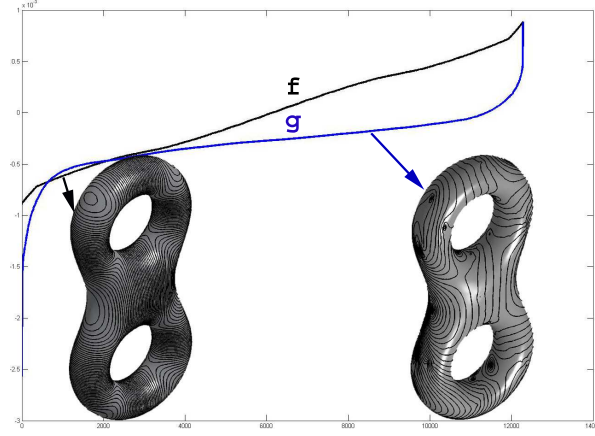
where the entries of the  $n \times n$  coefficient matrix  $\mathbf{A} := (a_{ij})$  are

$$a_{ij} := \begin{cases} \langle \nabla f(\mathbf{p}_i), \mathbf{w}_j \rangle & (i, j) \text{ is an edge,} \\ \sum_{j \in N(i)} \langle \nabla f(\mathbf{p}_i), \mathbf{w}_j \rangle & i = j. \end{cases}$$

We note that the structure of  $\mathbf{A}$  resembles the adjacency matrix of the triangle mesh; however, in our case  $\mathbf{A}$  is not symmetric and some entries might be negative. If we neglect degenerate cases where  $\text{rank}(\mathbf{A}) < n - 1$ , the unique solution of (3.8) is the vector  $\tilde{\mathbf{x}} = \tilde{\mathbf{1}}$ ; therefore, the only function orthogonal to  $f$  everywhere on  $D$  are the constant ones (see Figure 3.14).

To require that the orthogonality conditions (3.7) hold on the whole set of vertices has unique solution the constant functions. This result implies that two functions are everywhere orthogonal if only if one of them is constant. From the practical point of view, this result is quite unsatisfactory. Our idea is to relax the orthogonality constraints by requiring  $g(\mathbf{p}_i) \neq g(\mathbf{p}_j)$ , for at least two distinct indices  $i, j$ . More generally, we consider the problem: *given  $I \subseteq \{1, \dots, n\}$ , find  $g : D \rightarrow \mathbb{R}$  such that*

$$\begin{cases} \langle \nabla f(\mathbf{p}_i), \nabla g(\mathbf{p}_i) \rangle = 0 & i \in I^C, \\ g(\mathbf{p}_i) = \alpha_i & i \in I \end{cases} \quad (3.9)$$



**Figure 3.15:** The picture shows the co-domain and iso-contours of the function  $g$  orthogonal to a given  $f$  everywhere on  $D$  with the exception of the critical points of  $f$ .

where  $I^C$  is the complement of  $I$ . For  $i \in I^C$ , we rewrite (3.7) as

$$\begin{aligned} \sum_{j \in N(i) \cap I^C} \langle \nabla f(\mathbf{p}_i), \mathbf{w}_j \rangle g(\mathbf{p}_j) - g(\mathbf{p}_i) \sum_{j \in N(i)} \langle \nabla f(\mathbf{p}_i), \mathbf{w}_j \rangle &= \\ &= - \sum_{j \in N(i) \cap I} \langle \nabla f(\mathbf{p}_i), \mathbf{w}_j \rangle g(\mathbf{p}_j). \end{aligned} \quad (3.10)$$

If we assume that  $\sharp I = n - k$ , (3.10) is equivalent to the  $(n - k) \times (n - k)$  sparse linear system  $\mathbf{A}\mathbf{g} = \mathbf{b}$  where:

$$\mathbf{A} := (a_{ij}), \quad a_{ij} := \begin{cases} \langle \nabla f(\mathbf{p}_i), \mathbf{w}_j \rangle & i \in I^C, j \in N(I) \cap I^C \\ \sum_{j \in N(i)} \langle \nabla f(\mathbf{p}_i), \mathbf{w}_j \rangle & i = j \end{cases}$$

and

$$\mathbf{b} := (b_i)_i, \quad b_i := - \sum_{j \in N(i) \cap I} \langle \nabla f(\mathbf{p}_i), \mathbf{w}_j \rangle f(\mathbf{p}_j), \quad i = 1, \dots, n - k.$$

The sparse linear system (3.10) is efficiently solved by applying the conjugate gradient method [GV89]. Then, the hypothesis  $g(\mathbf{p}_i) \neq g(\mathbf{p}_j)$ ,  $i, j \in I$ , is enough to guarantee that  $g$  is not constant; clearly, if we set  $g(\mathbf{p}_i) = \alpha$ ,  $\forall i \in I$ , we achieve the solution  $g = \alpha$ . Once  $g$  has been calculated, the error on the orthogonality between  $f$  and  $g$  depends on the points of  $D$  where we did not impose the orthogonality condition and it is equal to  $\mathcal{I}(f, g) = \sum_{i \in I} \langle \nabla f(\mathbf{p}_i), \nabla g(\mathbf{p}_i) \rangle$ . Therefore, the angle between  $\nabla f(\mathbf{p}_i)$  and  $\nabla g(\mathbf{p}_i)$  will affect the orthogonality error measured by  $\mathcal{I}$ . Figure 3.15 shows the construction of its almost everywhere orthogonal function.

**Setting up the initial values.** A natural choice of  $I$  is the set of the critical points of  $f$  where the orthogonality conditions (3.9) is trivially satisfied. For each  $i \in I$ , let  $N(i)$  be its 1-star and  $\mathbf{n}_i$  be a user-defined vector (e.g., the vector orthogonal to the normal at  $\mathbf{p}_i$ ), then  $g(\mathbf{p}_i)$  is chosen as the minimum of the functional

$$G((\mathbf{p}_j)_{j \in N(i)}, \mathbf{p}_i) := \sum_{j \in N(i)} |g(\mathbf{p}_j) - g(\mathbf{p}_i)|^2$$

subject to the (discrete) linear constraints  $\nabla g(\mathbf{p}_i) = \mathbf{n}_i$ . If  $\sharp N(i) = k$ , the previous problem has  $(k + 1)$  unknowns and is efficiently solved by standard optimization techniques [GV89].

### 3.4 Discussions

The shape description framework we have introduced admits a flexible usage of real functions and computational techniques.

The Shape Graph we have defined in Section 3.1 may be tuned according to different user's needs and combines topological information with geometric one. In particular, it has been used to extend the computation of the size functions to surfaces and we have shown how the Shape Graph description can be used to code 3D scenes. Multi-dimensional size functions provide an innovative framework able to combine more properties in the same descriptor.

Moreover, we have proposed a method able to merge several descriptions induced by a family of functions defined on a surface  $D$  into a single representation, which is capable to make explicit the differences of the local properties measured by these mapping functions. We provided a direct relation between the critical points of two functions  $f, g$  through the study of the functional  $\mathcal{I}(f, g)$ . We have also demonstrated that, given a non constant real function  $f$ , the only functions that are orthogonal to  $f$  are constant unless the orthogonality condition is relaxed. Given  $\Gamma$ , we also showed how to calculate a new function  $g : D \rightarrow \mathbb{R}$  that is independent to  $\Gamma$  with respect to  $\mathcal{I}$  and we provided an efficient algorithm for its computation. As future work, we plan to use the proposed approach to study the evolution of time-depending functions defined on the same or several surfaces.

### Related publications

S. Biasotti. Topological coding of surfaces with boundary using Reeb graphs. *Computer Graphics and Geometry*, 7(1):31–45, 2005.

S. Biasotti and S. Marini. 3D object comparison based on shape descriptors. *International Journal of Computer Applications in Technology*, 23(2/3/4):57–69, 2005.

S. Biasotti, M. Marini, M. Spagnuolo, and B. Falcidieno. Sub-part correspondence by

structural descriptors of 3D shapes. *Computer Aided Design*, 38(9):1002–1019, September 2006.

S. Biasotti, D. Giorgi, M. Spagnuolo, and B. Falcidieno. Size functions for comparing 3D models. *Pattern Recognition*, 2008 (in print).

S. Biasotti, D. Giorgi, M. Spagnuolo, and B. Falcidieno. Size functions for 3D shape retrieval. In *SGP'06: Proceedings of the 2006 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, pages 239–242, 2006.

S. Biasotti, A. Cerri, P. Frosini, D. Giorgi, and C. Landi. Multidimensional size functions for shape comparison. Technical Report 04-07, IMATI, CNR, Genova (Italy), 2007.

A. Cerri, S. Biasotti, and D. Giorgi.  $k$ -dimensional size functions for shape description and comparison. In *ICIAP 2007: Proceedings of the 14<sup>th</sup> International Conference on Image Analysis and Processing*, Modena", September 10-14 2007. IEEE Computer Society Press.

L. Paraboschi, S. Biasotti, and B. Falcidieno. Comparing sets of 3D digital shapes through topological structures. In F. Escolano and M. Vento, editors, *Gbr2007: Proceedings of Graph-based Representations in Pattern Recognition*, volume 4538 of *Lecture Notes in Computer Science*, pages 114–125, Alicante, 2007. Springer Verlag.

L. Paraboschi, S. Biasotti, and B. Falcidieno. 3D scene comparison using topological graphs. In *Proceedings 5th Eurographics Italian Chapter Conference*, pages 87–93, Trento, 2007. The Eurographics Association.

S. Biasotti, G. Patané, M. Spagnuolo, and B. Falcidieno. Analysis and comparison of real functions on triangulated surfaces, In A. Cohen, J.-L. Merrien, and L. L. Schumaker, editors, *Curve and Surface Fitting*, pages 41–50. Nashboro Press, 2007.

---



## Chapter 4

# Applications and Results

The description framework introduced in Chapter 3 yields a high-level description of the surface shape that acts as investigation tool in several application fields. This framework guarantees both the computational efficiency and the topological coding of the object, which allows a qualitative and quick comparison of the object shapes. With respect to the previous works in this field, the proposed representations are able to faithfully represent the surface shape without distorting the semantic meaning of the theoretical descriptions while additional geometric attributes are added to yield the surface morphology.

In this Chapter we overview the application fields for which our representation framework has been successfully adopted. These concepts have been translated into the discrete domain and adopted shape matching [BMSF06, CBG07, BGSF06], scene comparison [PBF07b], shape classification [BGM<sup>+</sup>06, BGM<sup>+</sup>07] purposes and for best view selection [PPG<sup>+</sup>05]. For every task, prototypes based on libraries developed in *C* language have been implemented. A user interface of the functionalities of the descriptors has been built using the *Open Inventor* library [Wer94].

This chapter is organized as follows. In section 4.1, the effectiveness of using the *SG* structure for shape matching purposes is discussed. Results on shape retrieval using size functions (both 3D and multi dimension) are proposed. Then we shift to the partial matching problem, showing the effectiveness of the Shape Graph both for free-form and *CAD* models. The results of the application of the SG to scene comparison end Section 4.1. Section 4.2 shows how creative prototypes based on the Shape Graph may assist shape classification. Finally, in section 4.3 we propose the results obtained using our shape characterization to select a significant view of a 3D object.

## 4.1 Shape Matching and Retrieval

The problem of comparing shapes through their topological descriptors has been approached in a great number of ways. For example, several descriptors are based on the statistical distribution of the shape points in the space [VT03], spherical harmonic representations [KFR03], high-curvature regions [HK03], shape decomposition [DGG03], while others try to organize and interpret the shape features through a graph representation [ZTS02, HSKK01]. In particular, in this Section, we will focus on the efficacy of considering the shape descriptors, namely Shape Graph and high dimensional size functions, discussed in Chapter 3 to approach shape matching problems.

Depending on the choice the mapping function, the Shape Graph and size functions can guarantee a shape description suitable for shape comparison purposes. For instance, a suitable mapping function  $f$  has to be independent of rotation, translation, uniform scaling of the object and distribution of points on the surface. In addition, it is not admissible that there are vertices privileged a priori. These requirements prevent the use both of the height function [ABS03] and the centerline representation [LV99, HA03] for matching purposes, which respectively depend on the orientation and on the selection of a seed point. To be suitable for shape matching and retrieval issues any function  $f$  could be used, provided that it is invariant with respect to object rotation, translation and scaling [BMMP03, BMM<sup>+</sup>03b]. So, possible choices of the mapping functions are the distance from the barycenter, the average of the geodesic function [HSKK01], the geodesic distance from high curvature extrema [MP02], the shape elevation [AEHW06] and also families of functions, for instance the functions independent of the principal symmetry axis proposed in a [BCF<sup>+</sup>07].

As mentioned in Section 1.3, the most important aspect to evaluate when choosing the mapping function is the kind of features that we want to highlight in the description and the type of matching we would like to get. In our shape matching examples, the distance from the barycenter naturally highlights the distribution of the object with respect to its barycenter, like shown in figure 3.4. Therefore this function is rotation invariant, but sensitive to pose variations. On the contrary, the function in [HSKK01] is pose invariant because it depends on the shape distribution with respect to the geodesic center of the surface. In both cases, the shape will be described as a configuration of protrusions and hollows, but the geodesic will not discriminate between objects in different poses while the distance from the barycenter will do. Therefore, the geodesic is best suited for retrieving articulated objects disregarding the pose, while the distance from the barycenter will allow to distinguish among articulated models in different poses.

Also computational aspects have to be taken into account when choosing a mapping function. For example, the barycenter may be computed in linear time ( $O(n)$ ) with respect to the number of vertices and, since it depends on all surface vertices, it is robust to noise. On the contrary, the exact computation of the integral geodesic function may be performed only with



$O(n^2 \log(n))$  operations, where  $n$  is the number of vertices of  $S$ ; however, its approximation [HSKK01] runs in  $O(kn \log(n))$  operations, where  $k$  is a constant that represent a number of basis for the function evaluation. The approximation of the geodesic distance using the Dijkstra algorithm makes this function sensitive to the vertex distribution. To minimize the dependence of the geodesic function evaluation on mesh irregularity an uniform remesh operation is done on the models [AF06].

The tests on shape retrieval are performed mainly on the 280 triangle meshes classified in 14 classes of 20 models used in [BGM<sup>+</sup>06] and the benchmark of 400 triangle meshes used for the Watertight Track of SHREC (SHape REtrieval Contest) 2007 [GBP07]. In addition we have collected on the web and eventually remeshed 190 CAD models and created 160 object scenes.

The original models of our database were collected from several web repositories. Several models belong to the AIM@SHAPE repository [aim], the Princeton Shape Benchmark [pri], the 3D Cafe [3dc] web repository of 3D media, while several human models in different poses come from the CAESAR Data Samples [cae]. Most of the CAD models come from the National Design Repository at Drexel University [dre]. Moreover, we have downloaded the McGill 3D Shape Benchmark [mcg] that offers about 420 surface and volume models, classified in 19 classes.

#### 4.1.1 Shape retrieval using high dimensional size functions

In this Section we use our database to compare the retrieval performance of the approach proposed, with respect to three popular shape descriptors, namely the spherical harmonic descriptor<sup>1</sup> [KFR03], the view-based approach<sup>2</sup> [COTS03] and the Multiresolution Reeb graph [HSKK01].

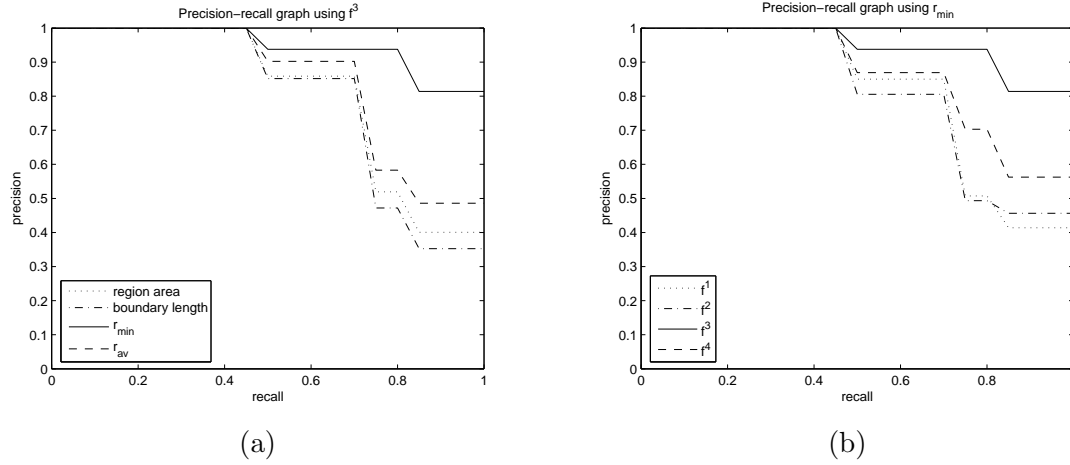
The results about size functions reported in this Section are obtained selecting, during a training phase, the most performing combination of features (mapping functions  $f$  and measuring functions  $\varphi$ ) to compute the size graph  $(G^f, \varphi)$ . The training phase has been performed using a smaller (20 items) database, containing a fraction of the models currently used as well as some additional models. The performance of different combinations has been evaluated in terms of the area of the region below the precision–recall curve, considering that larger areas indicate better results. In particular, the most performing pair  $(G^f, \varphi)$  is obtained when  $f$  is the integral geodesic distance, and  $\varphi$  the minimum radius  $r_{min}$  (see Section 3.2.1). Some comparative results are summarized in Figure 4.1. An intuitive explanation for these results is that the integral geodesic distance is suitable to deal with articulated objects, while the minimum radius of the surface regions provides an informative description of the local object shape, which is also stable to small perturbations.

---

<sup>1</sup><http://www.cs.jhu.edu/~misha/>

<sup>2</sup><http://3d.csie.ntu.edu.tw/~dynamic/3DRetrieval/index.html>

---



**Figure 4.1:** Precision-recall diagrams on the training dataset, involving different SG representations and attributes. (a) The mapping function to extract the centerline is the normalized integral geodesic distance [HSKK01]; the measuring function varies in a set of four attributes. (b) The mapping functions varies, while the measuring function is always the minimum radius of a region.

In what follows, we show some results on both the whole database and the single classes.

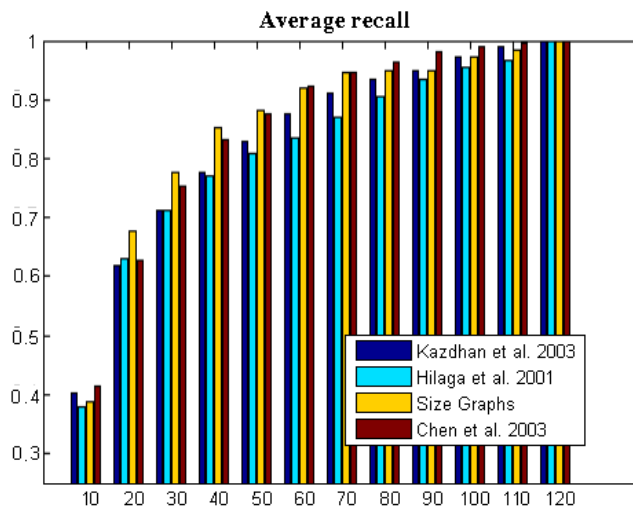
As a first performance parameter, we consider *percentage recall*. For a given number  $N$ , this parameter corresponds to the percentage of models in the same class of the query retrieved within the first  $N$  items. In particular, the recall histogram in Figure 4.2 are obtained computing percentage recall for the rank thresholds  $N = 10, 20, \dots, 120$ . Results are averaged over the whole database, and indicate that almost 80% of relevant items are retrieved within top 25% of the database (that is, within the first 30 models; remember that each class contains 20 elements). Moreover, in many cases the values obtained using size functions appears shifted up in the histogram, in comparison with respect to competitors, meaning a better performance.

Figure 4.3 (a) compares the *average rank* for the whole database obtained using size functions with the values obtained by the other techniques. The average rank is obtained by running each model of the database as a query and computing the retrieval rank of all members in the class of the query. The value obtained with size functions is the lowest one; notice that for this indicator lower values indicate better performance.

Another measure we use to assess the retrieval performance is the *last place ranking*, defined in [EBG98] as

$$L_n = 1 - \frac{\text{Rank}_l - n}{N - n},$$

where  $\text{Rank}_l$  indicates the rank at which the last relevant object is found,  $n$  is the number of relevant items and  $N$  is the size of the whole collection. The values computed are reported



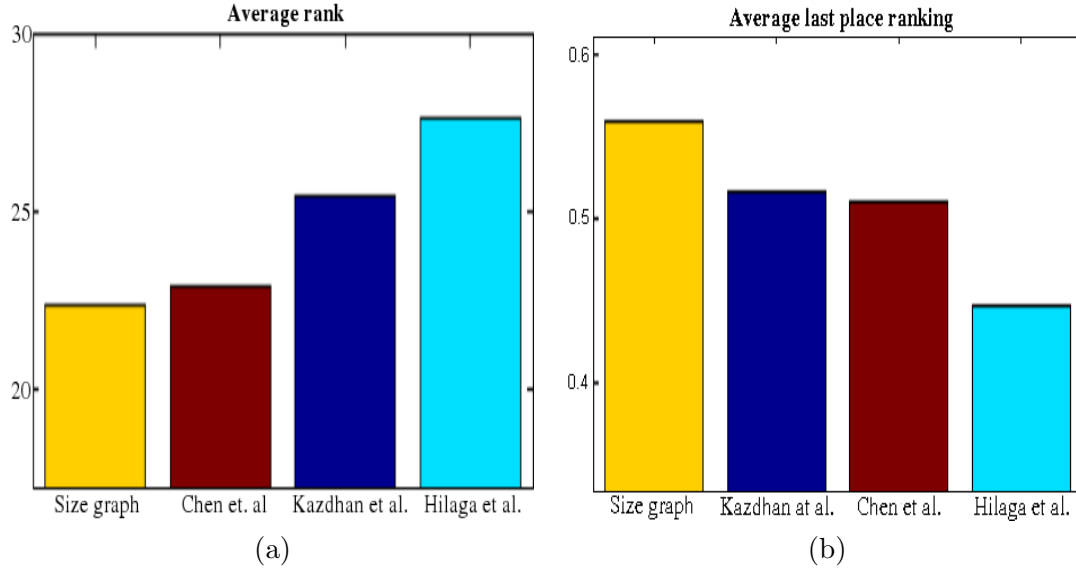
**Figure 4.2:** Comparison with existing retrieval methods. Recall histograms: the values are averaged on the whole database.

in Figure 4.3 (b). This measure gives an estimate of the number of items retrieved a user has to search in order to have a reasonable expectation of finding all relevant items. The higher this measure within the interval  $[0, 1]$ , the lower the number of items to check, meaning better results.

Finally, Figure 4.4 shows the standard precision–recall diagrams computed over the 5 classes in the database. The curves with different colors correspond to the different techniques we have tested. Remember that curves shifted upwards and to the right indicate a superior retrieval performance. For example, it is worth noticing the good performance realized by size functions for the classes of humans and glasses. These articulated models are well suited to enhance the features of our method, since the object in these classes share a common structure both for connectivity (e.g. a head, two arms and two legs for humans) and attributes (large sections near the barycenter, smaller ones for protrusions). The worst performance is realized when dealing with the class of animals with four limbs. The reason is that the models in this class are very heterogeneous. Our method takes into account, besides the structural properties – having four limbs – also the geometrical properties of the shape, which show too strong variations. In this particular case, better results could be obtained by considering a finer level of classification, rather than a basic one.

#### 4.1.1.1 Multi-dimensional size functions

Multi-dimensional size functions have been effectively applied to 2D and 3D triangulations and binary images. Given a model  $\Gamma$  represented by the geometric realization of a simplicial



**Figure 4.3:** Comparison with existing retrieval methods. (a) average rank and (c) last place ranking.

complex  $D$  of arbitrary dimension  $n$ , and a piecewise linear map  $\vec{\varphi} : \Gamma \rightarrow \mathbb{R}^k$ , we consider the size pair  $(\Gamma, \vec{\varphi})$ . Similarly, we consider also the size pair of a binary image with respect to a piecewise linear function.

Once the size pair has been identified, the proposed reduction of  $k$ -dimensional size functions to the 1-dimensional case allows us to use the existing framework for computing 1-dimensional size functions, see discussion in Section 3.2.2. In the application of the multi-dimensional size function framework to surfaces and volumes the size graph is

To effectively compute  $k$ -dimensional size functions, we have adapted the algorithm in [d'A00] to the half-planes of the foliation. In particular, for each half-plane we extract a *size graph*, i.e. an attributed graph, where each vertex is labelled by a real number.

For the simplicial case, our input is given by the 1-skeleton of the complex  $X$ , with the nodes labelled by the values of the restriction of  $\vec{\varphi}$  on the vertices of the complex; notice that, due to the piecewise linear nature of  $\vec{\varphi}$ , the size function of the geometric realization of the 1-skeleton coincides with the size function of the original size pair  $(|X|, \vec{\varphi})$ .

In case of digital spaces, the size graph corresponds to the graph used to encode the binary image. Similarly to the simplicial case, the node labels given in input correspond to the restriction function  $\vec{\varphi}$  over the vertices of the graph. Depending on the number of neighbors that may be adjacent to a point, the connectivity (i.e. the number of edges) of the size graph depends on the number of neighbors that are admitted to be adjacent to a point in a 2D image (i.e., 8-, 6- or 4- neighborhoods) or 3D image (e.g., 6-, 18- or 26-neighborhoods).

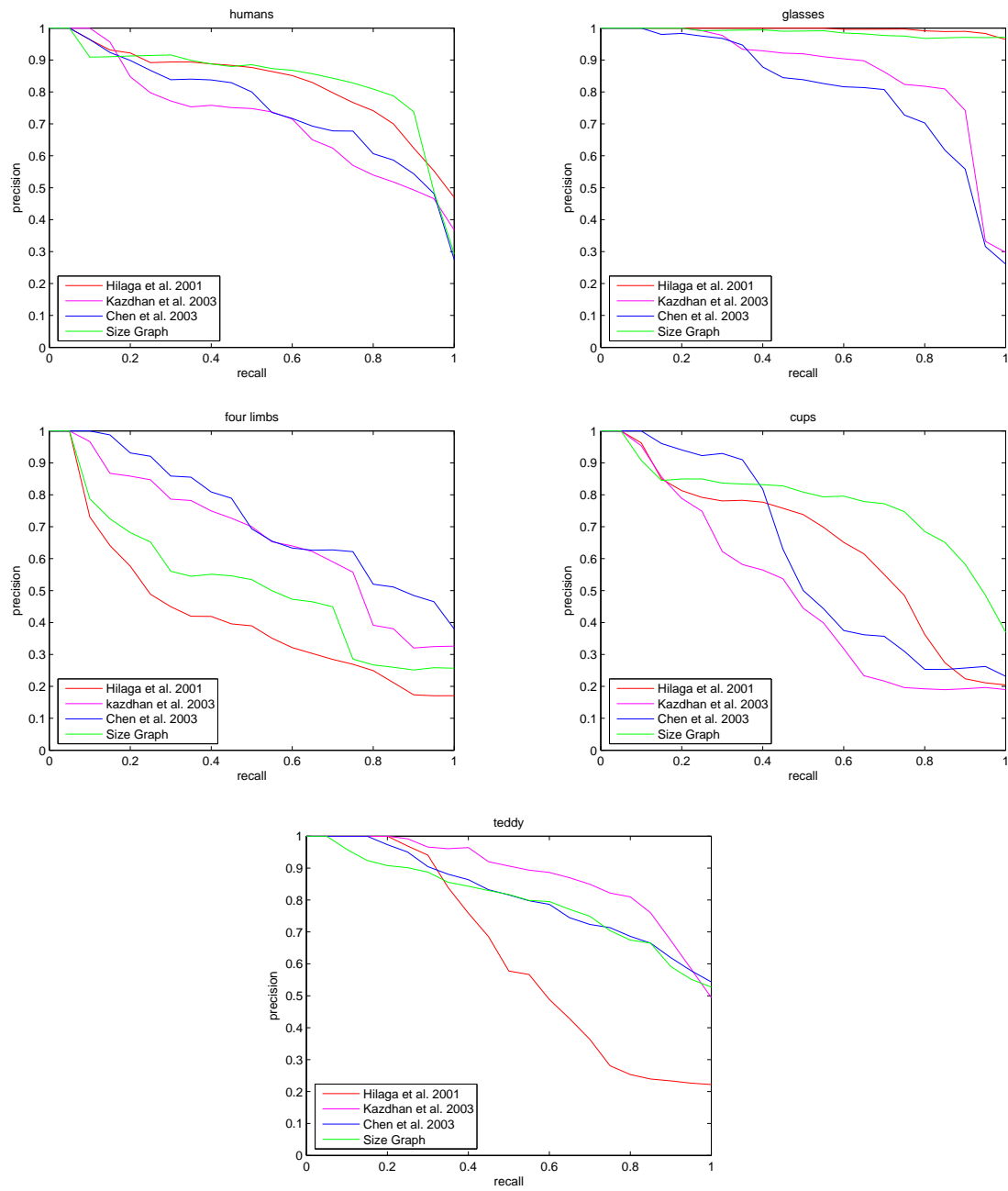


Figure 4.4: Precision-recall diagrams for the different classes in our database.

Evaluating the  $k$ -dimensional size function over a subset  $A \subseteq \text{Adm}_k$  of half-planes whose cardinality is  $\lambda$  means that we extract  $\lambda$  1-dimensional size functions, i.e., one for each half-plane of the foliation. Therefore, the computational complexity for evaluating the  $k$ -dimensional size function over  $\lambda$  half-planes is  $O(\lambda \cdot (n \log n + m \cdot \alpha(2m + n, n)))^3$ , where  $n$  and  $m$  are the number of vertices and edges in the size graph, respectively, and  $\alpha$  is the inverse of the Ackermann function [Ack28].

Denoting  $d(\ell_{(\Gamma, F_{(\vec{i}, \vec{b})}^{\vec{\varphi}})}, \ell_{(\Gamma', F_{(\vec{i}, \vec{b})}^{\vec{\psi}})})$  the (matching) distance [dFL05] between the 1SF's  $\ell_{(\Gamma, F_{(\vec{i}, \vec{b})}^{\vec{\varphi}})}$  and  $\ell_{(\Gamma', F_{(\vec{i}, \vec{b})}^{\vec{\psi}})}$  induced by the related representations by formal series, the distance between two kSF's  $\ell_{(\Gamma, \vec{\varphi})}, \ell_{(\Gamma', \vec{\psi})}$  can be defined as:

$$D(\ell_{(\Gamma, \vec{\varphi})}, \ell_{(\Gamma', \vec{\psi})}) = \sup_{(\vec{i}, \vec{b}) \in \text{Adm}_k} \min_{i=1, \dots, k} l_i \cdot d(\ell_{(\Gamma, F_{(\vec{i}, \vec{b})}^{\vec{\varphi}})}, \ell_{(\Gamma', F_{(\vec{i}, \vec{b})}^{\vec{\psi}})}).$$

Similarly to the approach adopted to compute  $k$ -dimensional size functions, the computation of the multi-dimensional matching distance easily follows from the computation of the 1-dimensional matching distance over the set  $A$  of half-planes. Since the computational complexity for computing the 1-dimensional matching distance in a single half-plane is  $O(p^{2.5})$ , with  $p$  the number of cornerpoints taken into account for the comparison, it follows that computing the  $k$ -dimensional matching distance between two  $k$ -dimensional size functions requires  $O(\lambda \cdot (p^{2.5} + k))$  operations.

Two types of 2-dimensional measuring functions have been considered in our experiments. First we have defined a 2-dimensional measuring function extending to triangle meshes and to the multi-dimensional case the reasonings in [LF02], where complete families of invariant 1-dimensional measuring functions are introduced. For a given triangle mesh of vertices  $\{P_1, \dots, P_n\}$  we compute the barycenter  $B = \frac{1}{n} \sum_{i=1}^n P_i$ , and normalize the model so that it is contained in a unit sphere. We then define a vector

$$\vec{v} = \frac{\sum_{i=1}^n (P_i - B) \|P_i - B\|^2}{\sum_{i=1}^n \|P_i - B\|^3}.$$

A parametric family of real-valued functions can be defined by setting, for each point  $P_i$  and for each  $\alpha \in \mathbb{R}$

$$\varphi_\alpha(P_i) = 1 - \frac{\|P_i - (B + \alpha \vec{v})\|}{\max_j \|P_j - (B + \alpha \vec{v})\|}.$$

We can now set:

$$\vec{\varphi}_{(\alpha_1, \alpha_2)}(P_i) = (\varphi_{\alpha_1}(P_i), \varphi_{\alpha_2}(P_i))$$

with  $\alpha_1, \alpha_2 \in \mathbb{R}$ ,  $i = 1 \dots n$ . The 2-dimensional measuring function we used in the experiments shown in Table 4.1 is  $\vec{\varphi} = \vec{\varphi}_{(1, -1)}$ .

---

<sup>3</sup>We recall from Section 2.3.1.2 that the computational complexity of the size function computation is  $O(n \log n + m \cdot \alpha(2m + n, n))$

---

It is worth noticing that the 2-dimensional measuring functions  $\vec{\varphi}_{(\alpha_1, \alpha_2)}$  are invariant with respect to translation and rotation. Moreover,  $\vec{\varphi}_{(\alpha_1, \alpha_2)}$  is well defined also if, for symmetry reasons,  $\vec{v}$  is the null vector. In this case the function  $\varphi_\alpha$  coincides with the distance from the baricenter for each value of the parameter  $\alpha \in \mathbb{R}$ . The invariance with respect to scale comes from the a priori normalization of the model.

We choose the foliation of  $\mathbb{R}^2 \times \mathbb{R}^2$  in half-planes  $\pi_{(\vec{l}, \vec{b})}$ , where  $\vec{l} = (\cos \theta, \sin \theta)$  with  $\theta \in (0, \frac{\pi}{2})$ , and  $\vec{b} = (a, -a)$  with  $a \in \mathbb{R}$ . In the examples, we consider the set  $A = \{(\vec{l}_i, \vec{b}), \quad i = 1, \dots, 17\}$  of admissible pairs, where  $\vec{l}_i = (\cos \theta_i, \sin \theta_i)$  with  $\theta_i = \frac{\pi}{36}i$ ,  $i = 1, \dots, 17$ , and  $\vec{b} = (0, 0)$ . For each half-plane  $\pi_i$  identified by  $(\vec{l}_i, \vec{b}) \in A$ , we compute the 1-dimensional measuring function  $F_{(\vec{l}_i, \vec{b})}^{\vec{\varphi}}$  defined in Theorem 3.2.1.

The values reported in Table 4.1 confirm the higher discriminatory power of 2-dimensional size functions with respect to the 1-dimensional case. Each cell of the table provides two values. The first one is the value of the 2-dimensional matching distance between the corresponding models. The second one is obtained computing the 1-dimensional matching distance between the 1-dimensional size functions associated to the 1-dimensional measuring functions  $\varphi_1$  and  $\varphi_{-1}$ , and taking the maximum value. For all the models in the dataset, the 2-dimensional matching distance produces a better lower bound for the 2-dimensional natural pseudo-distance (cfr. Theorem ). In other words, this example confirms that comparing 2-dimensional size functions furnishes a better approximation of the 2-dimensional natural pseudo-distance, with respect to comparing the single 1-dimensional measuring functions corresponding to the components of  $\varphi$ . Notice also that the values related to the 1-dimensional matching distance are each other very close, up to the considered number of digits, thus revealing a lower discriminatory power with respect to their concurrent use.

To perform shape retrieval experiments, we have selected a bi-dimensional measuring function  $\vec{\varphi} = (\varphi_1, \varphi_2)$  that does not belong to the family previously discussed. Here  $\varphi_1$  is a normalized Euclidean distance from the barycenter of the model, and  $\varphi_2$  is a normalization of the the averaged geodesic distance proposed in [HSKK01]. More formally,

















$$\varphi_1(v) = 1 - \frac{|v - B|_E}{\varphi_{1M}},$$

where  $B$  denotes the barycenter of the mesh and  $\varphi_{1M} = \max_{v_i \in V} |v_i - B|_E$ ; and

$$\varphi_2(v) = 1 - \frac{\sum_i g(v, b_i) \cdot \text{area}(b_i)}{\varphi_{2M}},$$

where  $g$  represents the geodesic distance,  $\{b_i\} = \{b_0, \dots, b_k\}$  is an almost uniform sampling of the vertices of the mesh,  $\text{area}(b_i)$  is the area of the neighborhood of  $b_i$  and  $\varphi_{2M} = \max_{v_j \in V} \sum_i g(v_j, b_i) \cdot \text{area}(b_i)$ .

In Figure 4.6 we depict the 2-dimensional size functions obtained combining  $\varphi_1$  and  $\varphi_2$ , restricted to three different half-planes of the foliation  $A$ . From left to right, for each model,

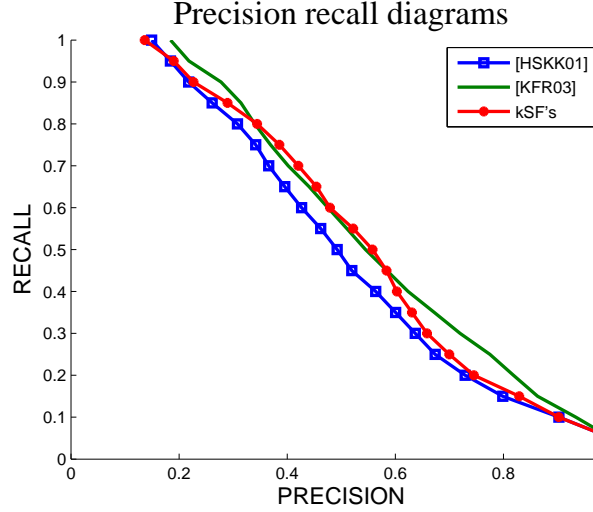
								
	0.0000 0.0000	0.0181 0.0003	0.1411 0.0025	0.1470 0.0026	0.1325 0.0023	0.1287 0.0022	0.1171 0.0020	0.1187 0.0021
	0.0181 0.0003	0.0000 0.0000	0.1419 0.0026	0.1478 0.0026	0.1304 0.0023	0.1265 0.0022	0.1171 0.0020	0.1187 0.0021
	0.1411 0.0025	0.1419 0.0025	0.0000 0.0000	0.0137 0.0002	0.1583 0.0028	0.1370 0.0024	0.1127 0.0020	0.1017 0.0018
	0.1470 0.0026	0.1478 0.0026	0.0137 0.0002	0.0000 0.0000	0.1533 0.0027	0.1381 0.0024	0.1137 0.0020	0.1021 0.0018
	0.1325 0.0023	0.1304 0.0023	0.1583 0.0028	0.1533 0.0027	0.0000 0.0000	0.0921 0.0014	0.0588 0.0016	0.1000 0.0017
	0.1287 0.0022	0.1265 0.0022	0.1370 0.0024	0.1381 0.0024	0.0921 0.0014	0.0000 0.0000	0.1069 0.0019	0.1048 0.0018
	0.1171 0.0020	0.1171 0.0020	0.1127 0.0020	0.1137 0.0020	0.0588 0.0016	0.1069 0.0019	0.0000 0.0000	0.0350 0.0006
	0.1187 0.0021	0.1187 0.0021	0.1017 0.0018	0.1021 0.0018	0.1000 0.0017	0.1048 0.0018	0.0350 0.0006	0.0000 0.0000

**Table 4.1:** Distances between models: the 2-dimensional matching distance between the 2-dimensional size functions associated to  $\vec{\varphi}_{(1,-1)}$  (top value) and the maximum between the 1-dimensional matching distances associated to the 1-dimensional measuring functions  $\varphi_1$  and  $\varphi_{-1}$  (bottom).

we represent the 1SF's that correspond to  $\vec{b} = 0$  and values of the angle  $\theta$  of  $\frac{\pi}{12}$ ,  $\frac{\pi}{4}$ , and  $\frac{5\pi}{12}$  respectively. Each row in the Figure 4.6 depicts, from left to right, a model and its 1SF associated to the three half-planes defined above. Observing the first two rows, we can notice how the same structure in size functions corresponds to the similarity between shapes. Indeed, the size functions of the two human models are each other very similar, while those of the third object are quite different. Moreover, these size functions homogeneously evolve over the half-planes of the foliation. Indeed the shape information conveyed by the multivalued measuring functions is distributed over the different half-planes. This means that the similarity (or dissimilarity) between objects can be evaluated by concurrently analyzing different shape properties. In other words, what we expect is that the  $k$ -dimensional size functions of similar objects are close one to the other over the whole foliation, thus guaranteeing also robustness with respect to small changes and perturbations of the model. Finally, Figure 4.5 shows that the retrieval performance of the bi-dimensional size function is comparable with other well known techniques [HSKK01, KFR03].

To analyze 3D binary images, we choose as set  $A$  of admissible pairs, the foliation of  $\mathbb{R}^3 \times \mathbb{R}^3$





**Figure 4.5:** Retrieval performance of our method over a database of 280 models, the MRG [HSKK01] and the Spherical Harmonics [KFR03].




given by  $\vec{l}_i = (\cos \theta \sin \phi, \sin \theta \sin \phi, \cos \phi)$ ,  $0 < \theta, \phi < \frac{\pi}{2}$  and  $\vec{b} = (0, 0, 0)$ . In our experiments, we consider the half-planes identified by  $a = b = 0$  and the following pairs of angles  $(\theta, \phi)$ :  $\{(\frac{\pi}{12}, \frac{\pi}{4}), (\frac{\pi}{6}, \frac{\pi}{4}), (\frac{\pi}{4}, \frac{\pi}{12}), (\frac{\pi}{4}, \frac{\pi}{6}), (\frac{\pi}{4}, \frac{\pi}{4}), (\frac{\pi}{4}, \frac{\pi}{3}), (\frac{\pi}{12}, \frac{5\pi}{12}), (\frac{\pi}{3}, \frac{\pi}{4}), (\frac{5\pi}{12}, \frac{\pi}{4})\}$ .

As measuring function we have chosen a 3-dimensional measuring function that respect the grid orientation of the voxels. In other words, we discriminate the models with respect to their spatial extent. Thus, denoting  $B = (B_x, B_y, B_z)$  the coordinates of the barycenter of the model, for each voxel  $v = (v_x, v_y, v_z)$  we have considered the 3D measuring function  $\vec{\varphi} = (\varphi_x, \varphi_y, \varphi_z)$ , where:

$$\begin{cases} \varphi_x = - |v_x - B_x|_E \\ \varphi_y = - |v_y - B_y|_E \\ \varphi_z = - |v_z - B_z|_E \end{cases}.$$

Table 4.2 highlights how also for 3D images the matching distance obtained over the half-planes of the foliations using more than one function improves the lower bound approximation of the natural pseudo-distance. We notice that the results contained in the fifth row of Table 4.2 provide the best approximation, over 9 half-planes selected in the foliation, of the k-dimensional matching distance.

in Sect. 3.2.2. The distances between six different objects in our database (two spiders, two cups and two manufactured models) are reported. These results are obtained using  $\vec{l} = (\frac{\sqrt{3}}{3}, \frac{\sqrt{3}}{3}, \frac{\sqrt{3}}{3})$  and  $\vec{b} = (0, 0, 0)$  as naive parameters to define an half-plane of  $\mathbb{R}^3 \times \mathbb{R}^3$ . As expected, the comparison framework satisfies the identity property, guaranteeing that a model has a null distance from itself. In addition, the distance between two models in the













		
$\theta = \frac{\pi}{4}, \phi = \frac{\pi}{4}$	0.1337	0.7593
$\theta = \frac{\pi}{4}, \phi = \frac{\pi}{12}$	0.1336	0.7592
$\theta = \frac{\pi}{12}, \phi = \frac{\pi}{4}$	0.2198	0.4741
average over 9 half-planes	0.1359	0.5449
maximum over 9 half-planes	0.2198	0.7593
$\varphi_x$	0.0782	0.2543
$\varphi_y$	0.1151	0.3481
$\varphi_z$	0.0712	0.0963

**Table 4.2:** Multidimensional and 1-dimensional matching distance between an airplane and another airplane (first column) or a chair model (second column). The first three rows refer to the multidimensional matching distance using 3 different singular half-planes of the foliation, the fourth and fifth row yield the average and the maximum value of the multidimensional matching distance using 9 planes, and the last three rows represent the 1-dimensional matching distance using the single components of  $\vec{f}$ .

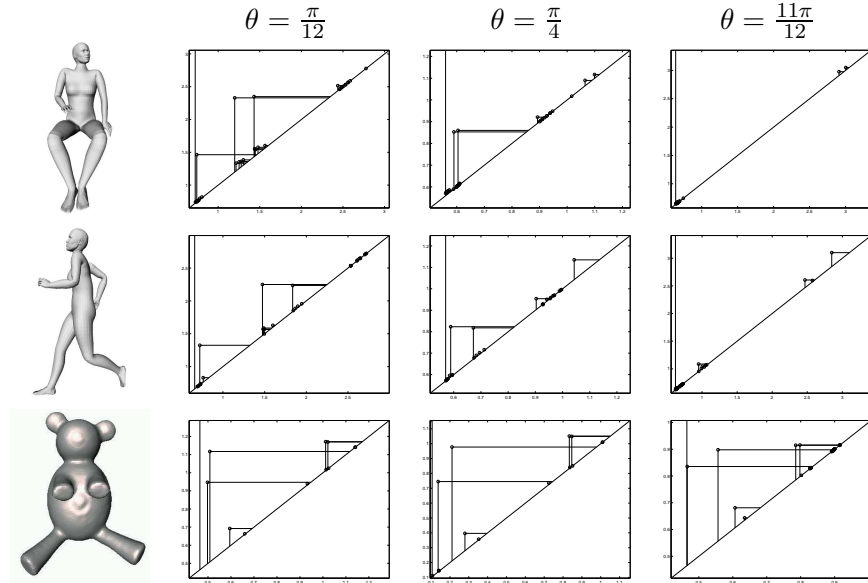
same class is significantly smaller than the distance between objects belonging to different classes (e.g. a spider and a manufactured model). We remark here that these distances may be used to rank the results of database queries and aggregate the models that share analogous properties, although they cannot be interpreted as "absolute" values, due to their dependence from the values of the measuring function and the foliation plane.

Finally, we show how the synergy of more shape properties, analyzed by means of multidimensional measuring functions, improves the recognition of the elements of a class. Figure 4.7 exhibits what happens when  $\varphi_x$ ,  $\varphi_y$ ,  $\varphi_z$  and  $\vec{\varphi} = \varphi_{xyz}$  are considered, either  $\varphi_x$ ,  $\varphi_y$ ,  $\varphi_z$  alone as 1-dimensional measuring functions (first three columns), or combined in a 3-dimensional measuring function  $\vec{\varphi}$  (last column). In addition, we compare the performance of  $\vec{\varphi}$  with respect to a 1-dimensional measuring function which is independent of the spatial embedding, namely the distance from the barycenter ( $\varphi_1$ ). In the columns of Figure 4.7 we rank the firstly retrieved items in the 3D image database [mcg], when the query is the model on the top row. It can be seen that the performance of  $\vec{\varphi}$  improves the retrieval results, diminishing the number of false positives. These results show that the  $k$ -dimensional size functions are promising, and we foresee their usage for retrieval of multidimensional digital shapes.

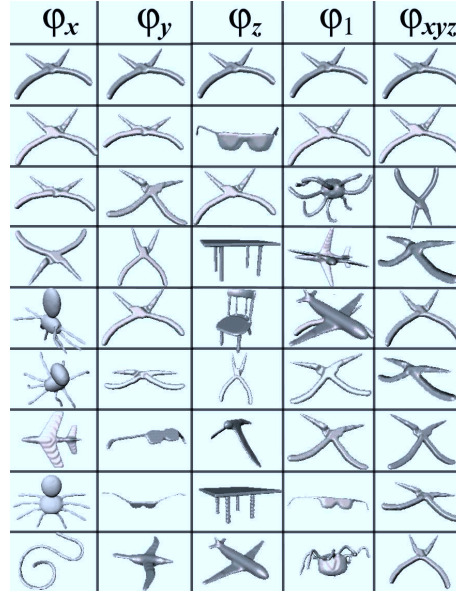
One of the attractive features of our approach is its flexibility, due to its geometric-topological nature and its capability to produce results which reflect human perception, by choosing the shape descriptors in a suitable way. In fact, the core idea of our method is the analysis of properties of real functions describing the shape under study. Real functions are involved for both the extraction of the Shape Graphs and the definition of the set of measuring functions, which serve for the computation of size functions. The role of the real functions

						
	0.00	1.59	7.77	7.77	23.54	23.99
	1.59	0.00	7.77	7.77	24.71	25.15
	7.77	7.77	0.00	3.42	22.63	23.08
	7.77	7.77	3.42	0.00	20.07	20.51
	23.54	24.71	22.63	20.07	0.00	1.25
	23.99	25.15	23.08	20.51	1.25	0.00

**Table 4.3:** Matching distances between six different models in our database over a single plane of a foliation. Computing  $420 \times 420$  comparisons between the models in the database requires 9.68s.













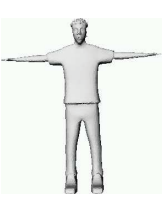
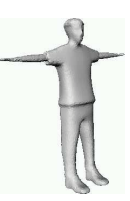






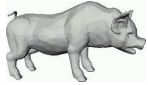





**Figure 4.6:** 1-dimensional size functions of three models over the half-planes of  $\mathbb{R}^2 \times \mathbb{R}^2$  defined by the parameters  $(\theta = \frac{\pi}{12}, \theta = \frac{\pi}{4}, \theta = \frac{11\pi}{12})$  and  $\vec{b} = (0, 0)$ .



**Figure 4.7:** Top retrieval results when four single measuring functions and the 3-dimensional size function that combines  $\varphi_x$ ,  $\varphi_y$  and  $\varphi_z$  are used. Results are depicted in every column in increasing order of distance from the first model.

is to take into account only the shape properties of the object which are relevant to the problem at hand, while disregarding the irrelevant ones, as well as to impose the desired invariance properties. Indeed, imposing invariance with respect to a transformation group simply means requiring the mapping and measuring functions to be invariant with respect to that group. Therefore, the added value of this approach to shape analysis relies on the possibility of adopting different functions as shape descriptors, according to the properties and invariants that one wishes to capture. When changing the functions, the resulting configurations can give insights on the shape from different perspectives. As a first example, if we aim to distinguish different positions in space, we should use mapping and measuring functions which emphasize the spatial distribution of the object shape, for example the distance from the barycenter, and the distances from the points on the bounding boxes. In Figure 4.8 (a) we see that, when searching for a standing-up human among the human models, their use produces a query response consisting of standing-up humans sharing a similar pose. Otherwise, if our idea is to put more emphasis, for example, on the “fatness” of the model rather than on its spatial pose, we can run the system selecting the region area as the measuring function. The results can be seen in Figure 4.8 (b), where the same query as before, a fat man, returns a fat man as the closer model, followed by a series of males. On the contrary, if we are interested in retrieving models showing a variety of poses, the integral geodesic distance reveals to be the best choice for the mapping function, since the centerline representation based on the integral geodesic distance does not distinguish, e.g., straight legs from legs with bent knees (see Figure 4.8 (c) where the measuring function is

$r_{av}$ ). Finally, in Figure 4.8 (d) the search for a pig in the class of animals with respect to the topological distance from curvature extrema and  $r_{max}$  produces a response that reflects both topological and geometric properties, according to perceptual similarity among the animals.

Query	1 <sup>st</sup> match	2 <sup>nd</sup> match	3 <sup>rd</sup> match	4 <sup>th</sup> match	5 <sup>th</sup> match
(a) 					
(b) 					
(c) 					
(d) 					

**Figure 4.8:** Query results according to different choices of the mapping and the measuring functions. (a) Emphasis on the spatial position, using  $f^1$  and  $\varphi_{P_5, P_6}$ ; (b) emphasis on the fatness, with  $f^1$  and the area of the regions; (c) humans in different poses are recognized, using  $f^3$  and  $r_{av}$ ; (d) results on the class of four-limbs reflecting perceptual similarity, obtained by  $f^4$  and  $r_{max}$ .

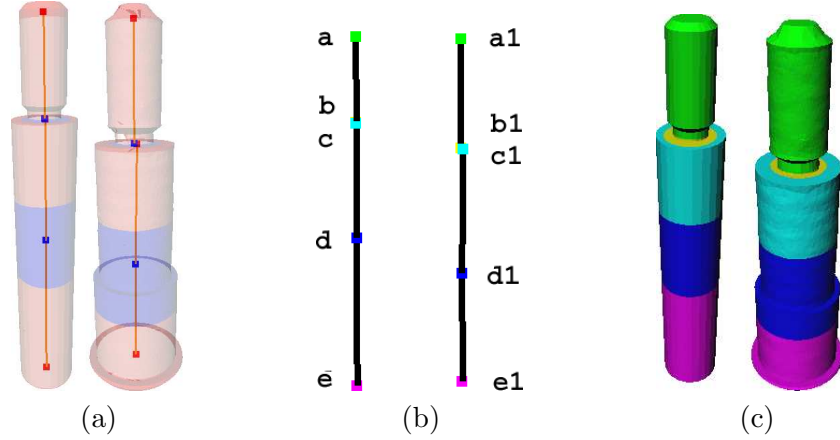
These results suggest that our approach could also be used as either a finer tool, after a rough filter has been used, or an instrument to refine queries in a retrieval pipeline, according to the user needs. The idea is that the interaction between the user and the system would allow tailoring the system response to the precise shape domain a user has in mind. This may be particularly true when considering our approach. While, in general, it may be difficult for the user to select the best combination of features to reach his goal, using our technique allows him to readily indicate the shape idea he has in mind, through the selection of a set of a features (i.e., mapping and measuring functions), which have a clear and intuitive geometric (and perceptual) significance.

### 4.1.2 Sub-part correspondence

The application of the shape graph to database retrieval has been discussed in the context of CAD models in [BM05]. The decomposition into significant regions induced by the SG defines a structural description of the shape, which is coupled with an error-correcting sub-graph isomorphism to build up a shape retrieval system. In particular, the proposed graph matching framework makes it possible to plug in heuristics for tuning the algorithm to the specific application and for achieving different approximations to the optimal solution. In [BMSF06] we discuss a method for measuring the similarity and recognizing sub-part correspondences of 3D shapes, based on the coupling of a structural descriptor, like the SG, with a geometric descriptor, like spherical harmonics. In the same paper it is discussed how a structure-based matching can improve the retrieval performance in terms of both functionalities supported (i.e., partial and sub-part correspondences) and the variety of shape descriptions that can be used to tune the retrieval with respect to the context of application.

As described in Section 3.1.1.2, the structural descriptor captures the relevant parts of the models and encodes them as nodes and edges of a graph. A uniform subdivision of the function image,  $[f_{min}, f_{max}]$ , is considered and the contour levels are inserted in correspondence of these function values. In particular the number of intervals in  $[f_{min}, f_{max}]$  is named *resolution* of the graph. The graphs associated to two objects are compared through the algorithm described in [BMSF06] and the correspondence between the nodes and edges that represent the common sub-graph is found. The common sub-graph (may be not connected) identifies the sub-parts of the objects that are similar, while the parts of the graphs that do not belong to the common sub-graph highlight the dissimilarities between the two models. The following convention is adopted to show the results: the sub-part correspondence between two models is represented by coloring the matched sub-part of each model with the same color, and different colors represent different connected components of the common sub-graph. When the common sub-graph corresponds to the whole graph of both the compared graphs, the color associated to each node reproduces the matched sub-parts of the two models as shown in figure 4.9.

We now provide and discuss some experimental results. Even if the partial matching framework we have proposed applies to generic 3D shapes, the experiments here proposed are mainly focused on the behavior of the descriptor and the comparison method on manufactured models. The experiments have been performed on about 250 models, collected into several web repositories. About 190 models, which comes from the National Design Repository at Drexel University [dre], have been uniformly remeshed to perform an accurate evaluation of the geodesic function. Other manufactured models have been considered from the AIMSHAPE repository [aim], from the Image-based 3D Models Archive [tsi] and the Princeton Shape Benchmark [pri]. Moreover, we have selected four classes: glasses, chairs, cups and tables of the database at the McGill 3D Shape Benchmark [mcg]. Since the original models were “voxelized” and seemed too rough, we smoothed them using a Laplacian



**Figure 4.9:** The two models and their graphs (a). The node correspondence is represented by nodes with the same color (b), and by regions of the same color.

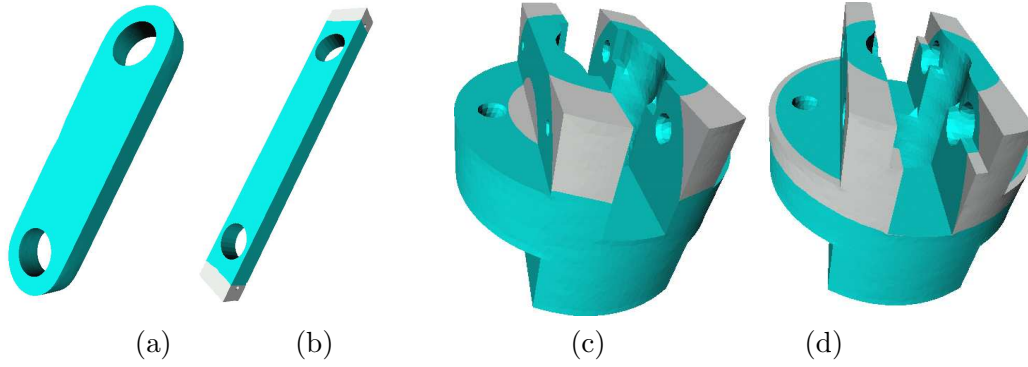
smoothing. Finally, to uniform the size of the models and make the results independent of scale operations, we uniformly scaled every model in a cube, centered in the origin of the Cartesian coordinates, whose edge length is two.

The experiments proposed in the following are performed and discussed for both sub-part correspondence and partial matching. The coupling of geometry and structure used by our descriptor naturally focuses on the partial matching problem; nevertheless the same method may be used for global matching proposed, like mentioned at the end of this Section. We emphasize that our matching framework performs most of the operations in a off-line pre-processing step and only search queries are computed on-line.

When discussing on partial matching, an important issue is how to evaluate the performance of the method. In fact, similar parts may belong to totally different models that belong to different families and also these cases must be recognized. These considerations highlight that precision-recall diagrams are not as significant as for the global matching problem.

In figure 4.10 the performance of the method for detecting the sub-part correspondence is tested. Although the overall shape of the models 4.10(a) and 4.10(b) (resp. 4.10(c) and 4.10(d)) is similar, the models differ for some small geometric details: the tips of 4.10(a) are rounded, while in 4.10(b) are squared; and the slots in the upper part of 4.10(c) differ from that of 4.10(d). In both cases, the partial matching process correctly recognizes the correspondence among similar sub-parts (light blue), while if some shape features have no correspondence in the other models, they are not mapped at all, as shown by the grey parts of the figure 4.10.

In figure 4.9 other two models with the same overall shape are shown. The models and their structural descriptors are shown in figure 4.9(a). The correspondence between the two graphs and the relative node mapping are shown in figure 4.9(b). Similar sub-parts of the



**Figure 4.10:** The sub-part correspondence between the models (a) and (b) is shown associating the same color to regions that are matched. Regions in grey are not matched. Similar results for (c) and (d).

two models are in figure 4.9(c). The differences between the two models are minimal but the two graphs describing them have an isomorphic structure. In this case the sub-parts corresponding to the nodes **c** and **c1** as well as **d** and **d1** are mapped even if they have small differences, but these differences can be recognized looking at the dissimilarity values, provided by the algorithm:

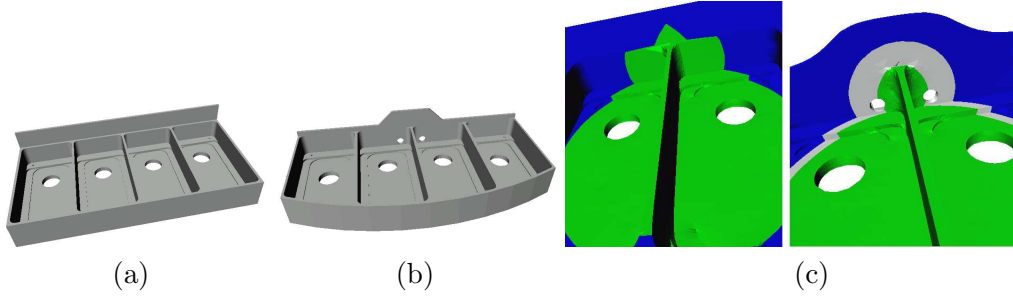
$$\begin{aligned}
 a &\longrightarrow a1 & : & 0.0308798 \\
 b &\longrightarrow b1 & : & 0.0131474 \\
 c &\longrightarrow c1 & : & 0.0559334 \\
 d &\longrightarrow d1 & : & 0.0988876 \\
 e &\longrightarrow e1 & : & 0.146666
 \end{aligned} \tag{4.1}$$

From (4.1) it is easy to perceive that the most dissimilar sub-parts correspond to the node pairs  $e \longrightarrow e1$  and  $d \longrightarrow d1$ .

Even though the global characteristic are analogous, the main difference between the two models shown in the figures 4.11 concern the two small holes in the rear part of 4.11(b) that are not present in 4.11(a). The comparison process recognizes the presence of the two holes and highlights the difference between the two models (figure 4.11(c)). The grey part (not colored) in the right part of figure 4.11(c) identifies the sub-part of 4.11(b) that is not mapped with any sub-part of 4.11(a). The comparison process between the two structural descriptors produces a non connected common sub-graph made of two connected components separated by the non matched sub-part (in grey). In figure 4.11(c), the two connected components are shown with different colors (blue and green).

Other examples are shown in figure 4.12. In 4.12(a) it is highlighted how the front and the back parts of two chairs correspond, while colors in figure 4.12(b) show the sub-part correspondence of two articulated models whose descriptors have the same structure. In





**Figure 4.11:** The model (b) has two small holes on its rear part that do not have any counterpart in the model (a). A detail of the correspondence between sub-parts the two models is shown in (c): the grey region identifies the difference between the two models.

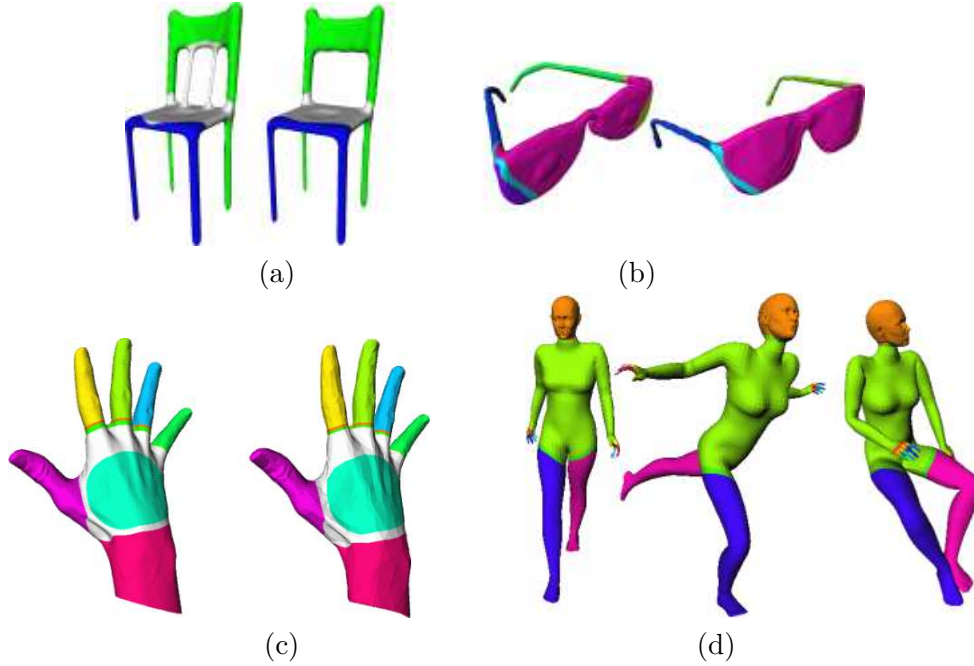
figure 4.12(c) two hands (hand-273k and hand-3k) are compared. The different number of triangles in the two models (see table 3.1) is the cause of the lack of correspondence between the palm and the rest of the hand. Finally, figure 4.12(d) represents the subpart correspondence of model representing a woman in three different poses.

An example of partial correspondence among models having different overall appearance is given in figure 4.13. In this case the models differ both in structure and geometry but they have similar sub-parts: even if the two big protrusions are not similar from a geometric point of view they have similar structure. The coupling of geometry and structure pursued by our algorithm correctly map these sub-parts. On the contrary, the central parts of the models are different both in structure and geometry, thus they are not recognized as similar sub-parts.

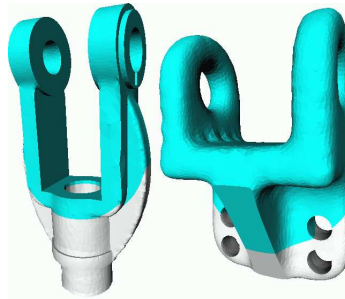
Once the similar sub-parts have been recognized, the node mapping between the two structural descriptors automatically provides a way for mapping the model sub-parts associated to the graph nodes. In figure 4.14(a) the two models of figure 4.13 are shown along with their graphs, while in figure 4.14(b) the two graphs has been matched and the node and edge correspondence is represented. According with such a correspondence, the figure 4.14(c) shows the more detailed sub-part correspondence.

Figure 4.15 shows two examples where a whole model is a sub-part of another one. Even if the two shapes of 4.15(a) and 4.15(b) are dissimilar, the elongated part shown in 4.15(b) is correctly recognized as sub-part of the model 4.15(a), because the structural descriptor of 4.15(a) correctly characterizes both the geometry and the structure of the sub-part that identifies the long part of the screwdriver. The same happens for the models 4.15(c) and 4.15(d). In this case, the sub-part containing the four holes of the model 4.15(c) is not mapped to any part of the model 4.15(d). The detail highlighted in the square (figure 4.15(c)), shows that the topology is preserved and the holes do not belong to the shape correspondence.

Other examples concerning the sub-part correspondence between 3D models are shown in

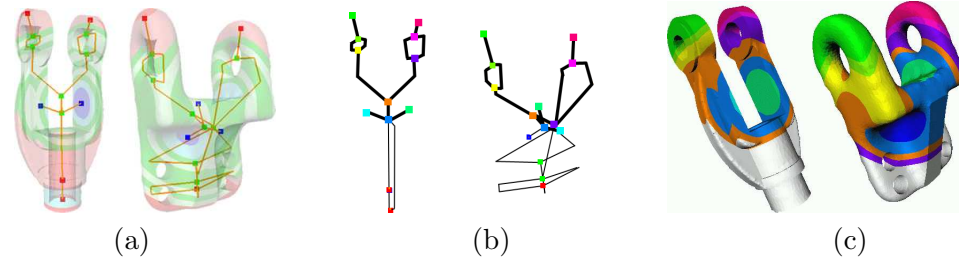


**Figure 4.12:** Correspondence between the sub-parts of two chairs (a), two glasses (b), two hands (c) and a human model in three different poses (d).

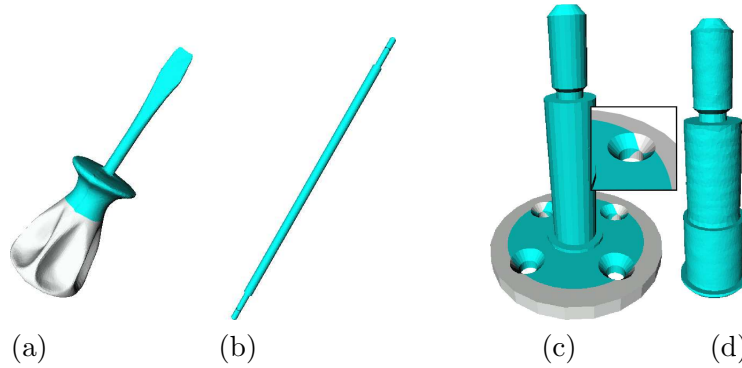


**Figure 4.13:** The correspondence between the sub-parts of the two models is highlighted by the color.

figure 4.16. The cubes in figure 4.16(a) have two blind-holes with different shape and the structural descriptor encodes these differences and the comparing algorithm highlights them; the left model in 4.16(b) has a deep squared pocket on one of its faces, while the blend region in the two models in 4.16(c) significantly differ. In both the cases the similar and the dissimilar sub-parts are correctly recognized. The two springs of figure 4.16(d) have different spatial distribution but the structure is the same, as well as models in 4.16(e), 4.16(f) and 4.16(g). The models in 4.16(h) have the same structure but the sub-parts have different proportion. This produces a difference both in structure and in geometry.



**Figure 4.14:** Two models are shown together with their graphs (a). The correspondence between the two graphs is represented by nodes with the same colors (b). The node correspondence between the two graphs is reproduced on the two models, respecting the colors associated to the nodes (c).

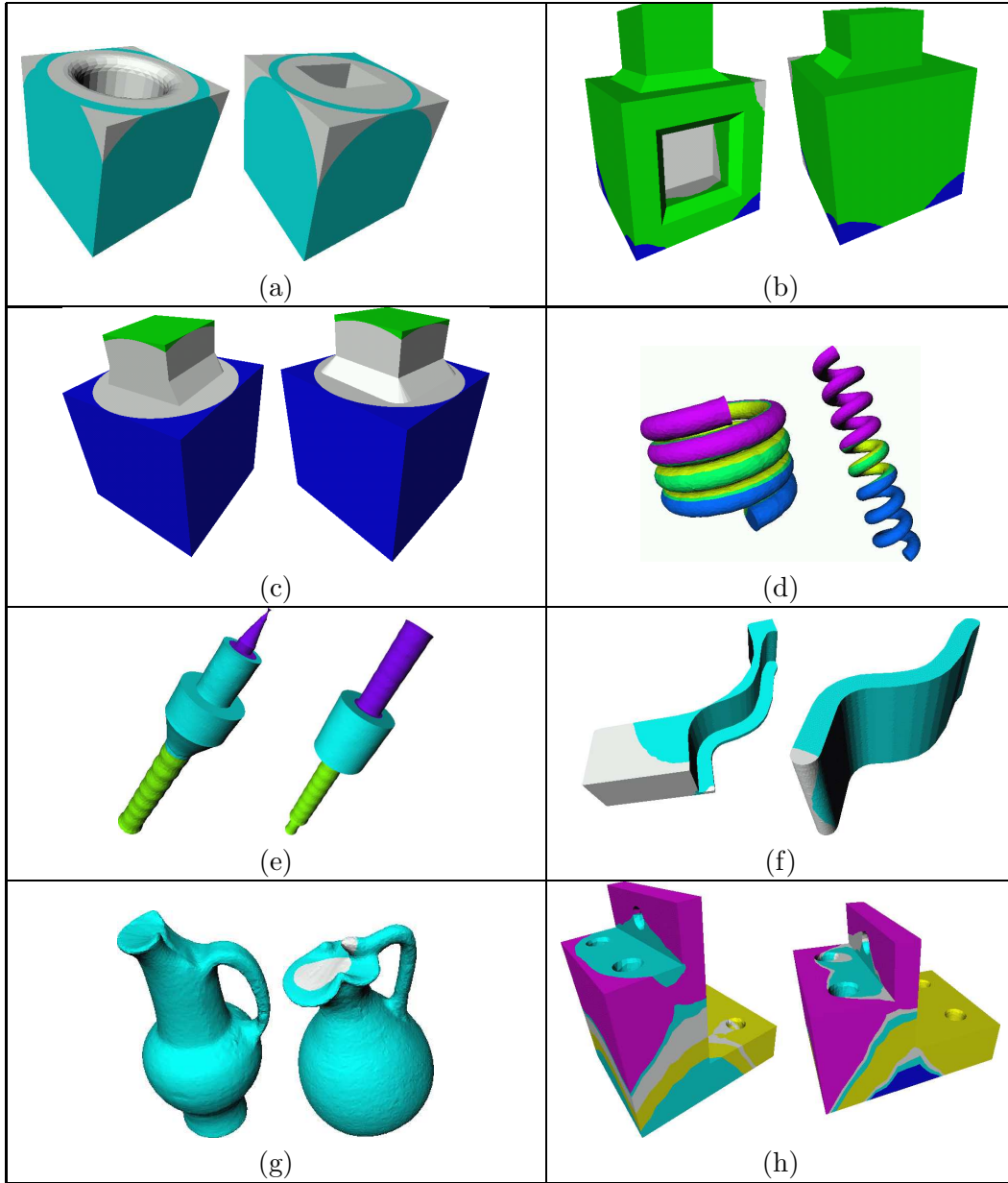


**Figure 4.15:** The models (b) and (d) are recognized as sub-parts of the models (a) and (c) respectively.

Finally, we report the results of our Shape Graph on the Partial Matching Track over Watertight Models of the SHape REtrieval Context 2007 [MBP07]. The dataset used in the experiments consists of 400 closed, manifold triangle meshes grouped into 20 classes made of 20 objects, see Figure 4.17.

A set of 30 query models has been built combining subparts of models belonging to the dataset. The whole set of queries is shown in Figure 4.18. To evaluate the performance of a method, a set of *highly relevant*, *marginally relevant* and *non-relevant* models belonging to the dataset has been associated to each query model.

As performance indicator we adopt the *Normalized Discounted Cumulated Gain vector* (NDCG), [JK02]. The computation of the NDCG is based on a gain vector, which is obtained by the ranked list where the model's identifiers are substituted with their "relevance scores" and where the relevance scores depend on the definition of the ground-truth. In our implementation, the weights of highly relevant, marginally relevant and non-relevant models are 2, 1 and 0 respectively. Then, a *Discounted Cumulated Gain*, (DCG) vector is recursively defined as:

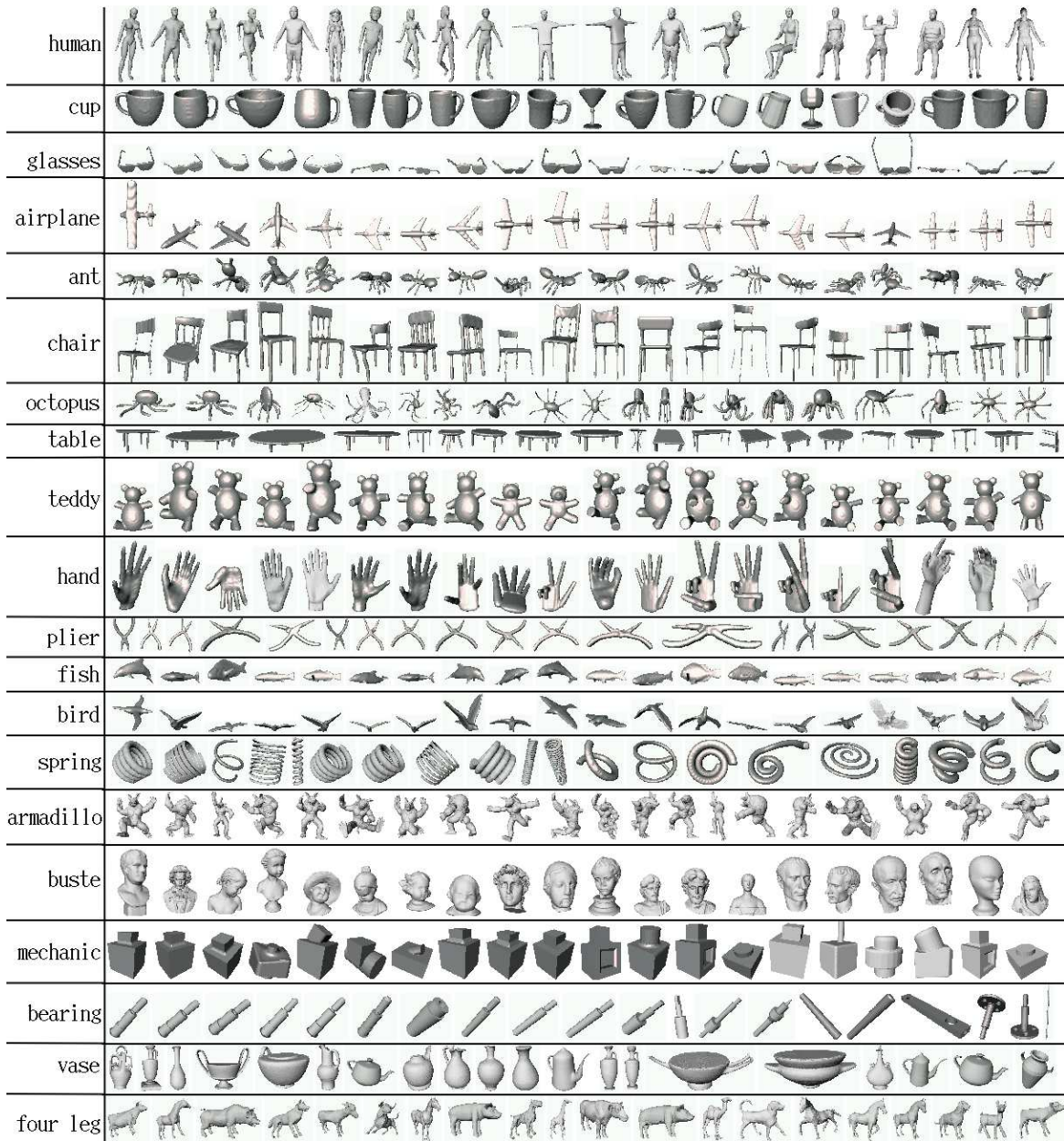


**Figure 4.16:** *Some examples of partial correspondences.*

$$DCG[i] = \begin{cases} G[i] & \text{if } i = 1, \\ DCG[i-1] + (G[i]/\log i) & \text{otherwise,} \end{cases}$$

where  $G[i]$  represents the value of the gain vector at the position  $i$ . The normalized dis-

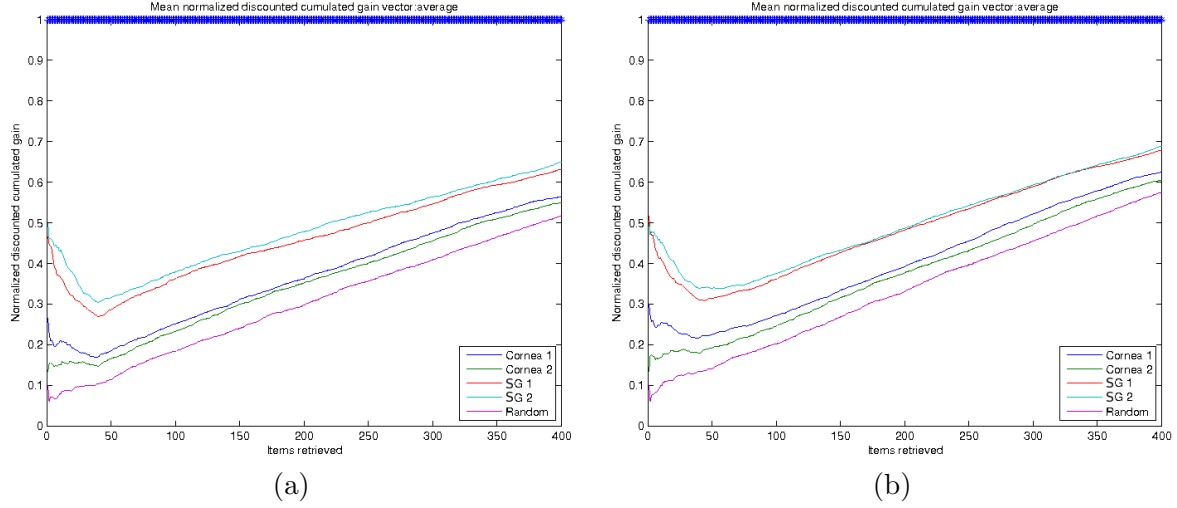
---



**Figure 4.17:** (a) SHREC database, containing 400 models divided into 20 classes. Each row represents a class of objects.







**Figure 4.19:** Normalized Discount Cumulated Gain considering only highly relevant models (a) and both highly relevant and marginally relevant models (b).

A *feature vector* of  $G$  is defined as the matrix  $\mathbf{B}$  such that:

$$\mathbf{B} = (f_{1,1}, \dots, f_{1,n}, \dots, f_{n,1}, \dots, f_{n,n})^t$$

where  $f_{i,j} = \text{sign}(S_j(\Phi_{1,i}, \dots, \Phi_{n,i})) \ln(1 + |S_j(\Phi_{1,i}, \dots, \Phi_{n,i})|)$  are elements of a matrix  $F = (f_{i,j})_{i,j=1,\dots,n}$ .

Then, the distance between two graphs  $G_1, G_2$  whose feature vectors  $B_i, i = 1, 2$ , are known, is defined by:

$$d(G_1, G_2) = \|\mathbf{B}_1 - \mathbf{B}_2\| . \quad (4.2)$$

We have modified the distance 4.2 to deal with the Laplacian matrix that represents a set of SGs built according to the rules explained in Section 3.1.2. In particular to compare two scene graphs (for simplicity we still denotes the graph as  $G_1^f$  and  $G_2^f$ ), we introduce a new distance:

$$D(G_1, G_2) := \left| \|\mathbf{B}_1\|_2 - \|\mathbf{B}_2\|_2 \right| . \quad (4.3)$$

$D$  is a pseudo-metric: it satisfies positivity, symmetry and triangle inequality; identity is not verified ( $D(G_1, G_2) = 0 \not\Rightarrow G_1 \simeq G_2$ ). Between  $D$  in (4.3) and  $d$  in (4.2) the following relation holds:

$$D(G_1, G_2) = \left| \|\mathbf{B}_1\|_2 - \|\mathbf{B}_2\|_2 \right| \leq \|\mathbf{B}_1 - \mathbf{B}_2\|_2 = d(G_1, G_2) .$$

In our experiments we use a normalized version of (4.3), which is useful to arrange a database with  $m$  sets; let it be  $\|\mathbf{B}_i\|_2 = \max_{i=1,\dots,m} \|\mathbf{B}_i\|_2$ ; then

$$D_m(G_1, G_2) = \frac{|\|\mathbf{B}_1\|_2 - \|\mathbf{B}_2\|_2|}{\|\mathbf{B}_i\|_2} . \quad (4.4)$$

(4.4) has the same properties of  $D$ , that is, it is a pseudo-metric.

Finally, we propose also another normalization of  $D$  (see [PBF07a]):

$$D_N(G_1, G_2) = \frac{|\|\mathbf{B}_1\|_2 - \|\mathbf{B}_2\|_2|}{\max(\|\mathbf{B}_1\|_2, \|\mathbf{B}_2\|_2)};$$

$D_N$  is a pseudo-semi-metric, that is, it satisfies neither identity nor the triangle inequality, and it is suitable to compare just two scene graphs. Moreover, if we define  $d_N(G_1, G_2) = \frac{\|\mathbf{B}_1 - \mathbf{B}_2\|_2}{2 \max(\|\mathbf{B}_1\|_2, \|\mathbf{B}_2\|_2)}$  as a normalization of (4.2), it follows that  $D_N(G_1, G_2) \leq 2d_N(G_1, G_2)$ , and therefore our measure is a lower bound of the distance proposed in [WHL05].

We discuss now the stability and the robustness of the measures we have introduced.  $\|\mathbf{B}\|_2$ , and consequently  $D_m$ , is well-conditioned. In fact, it is stable with respect to graph perturbations. For example, if a class of the database used is chosen and each model attribute is slightly perturbed (1% or 2%), the retrieval performance of the method is unaltered. Moreover, if the distance between a graph and its perturbed one is considered, it happens that  $D_m$  is lower than 0.08 in the 90% of cases.

As far as robustness is concerned, an inherent numerical error appears in computing symmetric polynomials, that is  $S_j(\Phi_{1,i}, \dots, \Phi_{n,i}) = \lambda_i^{\frac{j}{2}} \sum_{k_1 < \dots < k_j} V_{k_1,i} V_{k_2,i} \dots V_{k_j,i}$ . Actually it is typical when working with Laplacian matrices that the most significant information on graphs is related to the first eigenvalues, and a common practice is to eliminate some of the last ones. In our context, to guarantee coherent results, it is necessary discarding some eigenvalues: since the last ones are significantly bigger than the previous, a numerical error appears and distorts results. Since the growth of the matrix spectrum has been shown to be almost linear ([DBG<sup>+</sup>06]), the biggest eigenvalues can be automatically removed analysing the increase of the spectrum and discarding the values that diverge from linearity.

Four different mapping functions  $f$  are considered in the scene comparison framework, namely the distance from the barycenter, the height function (with respect to  $z$  axis), the integral geodesic distance, originally proposed by Hilaga et al. [HSKK01], and the minimum distance from a source point ([LV99]).

For every node  $v \in V$  associated to a region  $R$ , we use the geometric attributes  $\varphi(R)$ , namely the minimum, maximum and average distance of the barycenter of  $R$  from the region vertices, the sum of the lateral areas for each component of  $R$ , the sum of  $B_R$  lengths and the value of  $f$  in every vertex in  $V$ . Details on this representation can be found in of Section 3.1.1.1 and [BGSF08a].















Let us consider two scenes with a human and an animal model (Figure 4.20): they look really similar, in particular the human model stands up in both scenes. In this case the height function could be a good compromise between representation and computational efficiency.

$D_N$	Attributes
0.0122	minimum distance
0.0053	maximum distance
0.0226	average distance
0.0091	sum of the pseudo-cone areas
0.0267	sum of $B_R$ lengths
0.0462	value of $f$
0.0204	average distance

**Figure 4.20:** Scenes with a man and a teddy bear: distance values when using height function

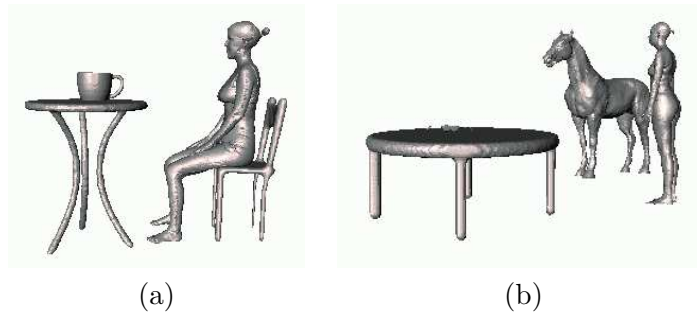
The results in Figure 4.20 are consistent with our perception: in fact the biggest distance is 4.6%, which is a low value for  $D_N$ . On the contrary, if we chose the height function with respect to another direction (for example, the  $x$  axis), we would not observe similarity between the two scenes. Similarly, comparing some chairs analysed again through the height function we obtain the average distances in Figure 4.21. The underlined values refer to the fallen chair: as we can imagine, these distances are bigger than the others, because the used function is sensitive to the position of the object in the reference space.

						
	0	0.1959	<u>0.1217</u>	0.1016	0.1293	0.2392
	0.1959	0	<u>0.2536</u>	0.1057	0.0763	0.0526
	<u>0.1217</u>	<u>0.2536</u>	0	<u>0.1854</u>	<u>0.1969</u>	<u>0.2699</u>
	0.1016	0.1057	<u>0.1854</u>	0	0.0344	0.1536
	0.1293	0.0763	<u>0.1969</u>	0.0344	0	0.1261
	0.2392	0.0526	<u>0.2699</u>	0.1536	0.1261	0

**Figure 4.21:** Scenes with one chair

We show a simple example to explain how the distance  $D_N$  behaves when comparing two

sets of objects ([PBF07a]). The two scenes, and their subscenes, in Figure 4.22 (a) and (b), are analysed comparing the scene graphs provided by the barycenter distance function. In the first scene, there is a woman sitting on a chair in front of a table, on which there is a cup, while in the second one there are a table and a pair of spectacles on it, and next a woman and a horse. When the sum of the pseudo-cone lateral areas is considered as node attribute, the distance is  $D_N = 0.2276$ , which is a quite high result. On the contrary, if we focus on the woman and the table in both scenes, the distance  $D_N = 0.0074$  indicates a possible, maybe strong similarity. Again, when the table and the cup in (a) are compared with the table and the spectacles in (b),  $D_N = 0.1617$  refers to less similar subscenes. Finally, if we analyze together the table and the woman in (a) with the table and the horse in (b),  $D_N = 0.0894$ : this result is an example of *false positive* (that is, a false result of similarity), and is probably due to the fact that the Shape Graph of the chair is structurally similar to that of the horse.

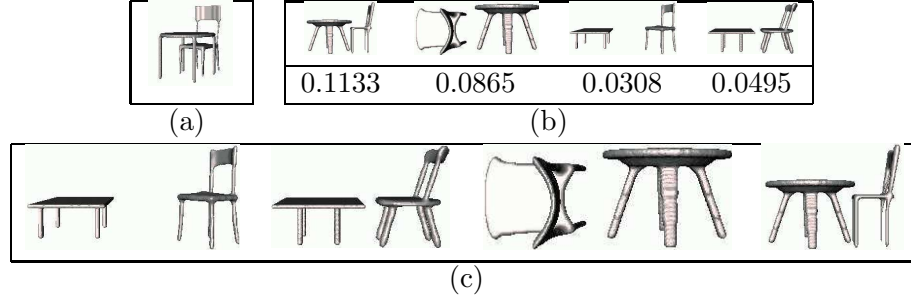


**Figure 4.22:** The values of the distance  $D_N$  between these two scenes is generally high; this means that the two scenes are not globally similar.









As a further example, we show a retrieval result (Figure 4.23), where the family of scenes with a chair and a table is re-ordered with respect to the query  $Q$  (attribute: average distance of the barycenter from the region vertices). Since the distances we obtain are those in Figure 4.23(b), then ordering the database with respect to  $D_N$  we get Figure 4.23(c). This is a consistent result, because, for example, in the third and fourth scene there is the same table, which is quite different, being thicker than the other ones.

Let us now observe the four scenes in Figure 4.24: again, they are composed by the same kind of models, however we can expect a bigger similarity between the third and the fourth, in which the man has open arms. For every function we report the average distances : height ( $h$ ), barycenter distance ( $b$ ), the minimum distance from a source point ( $mds$ ) and the integral geodetic distance ( $igd$ ).

With respect to every function, the biggest similarity is between the third and the fourth scene (double-marked square); moreover, leaving  $igd$  results out, each distance remains below 13%. As established a posteriori, it seems to be a result of similarity. This example shows us that the integral geodetic distance cannot guarantee a reliable result, and it is not so suited for our purpose.





**Figure 4.23:** (a) Query, (b) the database and the distances from  $Q$ , and (c) the database ordered according to the distance

				
	0	0.1028( <i>h</i> ) 0.0478( <i>b</i> ) 0.0571( <i>mds</i> ) 0.2026( <i>igd</i> )	0.0232( <i>h</i> ) 0.0817( <i>b</i> ) 0.0774( <i>mds</i> ) 0.3662( <i>igd</i> )	0.0177( <i>h</i> ) 0.0977( <i>b</i> ) 0.0903( <i>mds</i> ) 0.4292( <i>igd</i> )
		0	0.1236( <i>h</i> ) 0.0355( <i>b</i> ) 0.0715( <i>mds</i> ) 0.2120( <i>igd</i> )	0.1187( <i>h</i> ) 0.0523( <i>b</i> ) 0.0531( <i>mds</i> ) 0.2894( <i>igd</i> )
			0	0.0111( <i>h</i> ) 0.0175( <i>b</i> ) 0.0413( <i>mds</i> ) 0.1448( <i>igd</i> )
				0

**Figure 4.24:** Scenes with a man and a table

Finally, in Figure 4.25 we show the distances between a chair and a horse whose graphs are obtained through barycenter distance with respect to every attribute and then the average distance. This last value is 0.094, which could indicate similarity. Obviously a chair is not similar to a horse, while their graphs are comparable: it is an example of a so called “false positive”.

We arrange the experiments in a scene repository as follows: first of all we extract the descriptors of the scene components, and then estimate the distance between two scenes using (4.4): the smaller  $D_m$  is, the more the two scenes, or better, their structures, are similar. We have generated and preclassified two databases considering in the same class two scenes if they have the same number of objects and also the same kind of components (for example, in a class there are scenes with a human and an animal, in another one a

		
	0.0987	1 <sup>st</sup> attribute
	0.0982	2 <sup>nd</sup> attribute
	0.0907	3 <sup>rd</sup> attribute
	0.1516	4 <sup>th</sup> attribute
	0.0963	5 <sup>th</sup> attribute
	0.0282	6 <sup>th</sup> attribute
	0.0940	global average
		

**Figure 4.25:** A false positive result

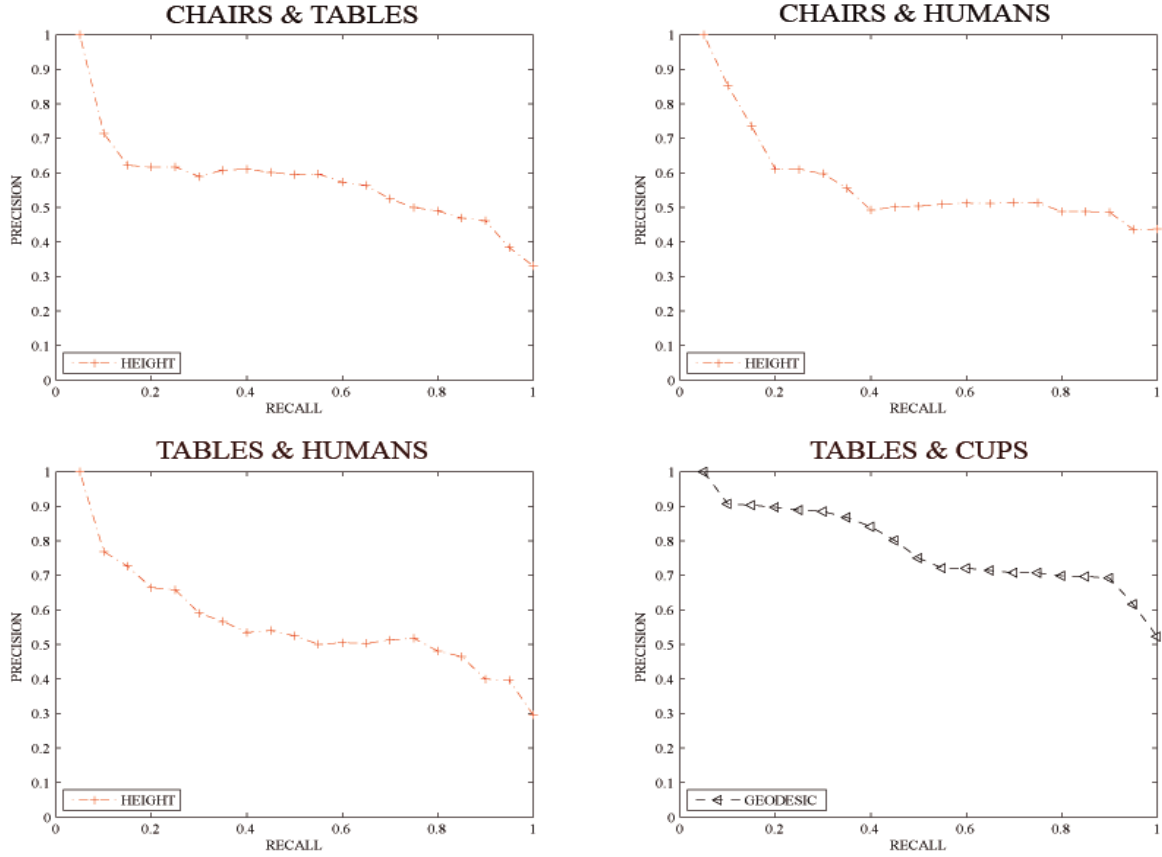
human next to a chair). According to these assumptions, *Data1* consists of 4 scene classes each of them with 20 scenes: in the first class there are a table and a chair, in the second a chair and a human, in the third a table and a human, and in the last a table and a cup. In Figure 4.26 we present the best precision-recall diagram for each class. This choice is due to the extremely large possibilities that we have when comparing graphs. In this sense, let us remember that, in our context, each object graph is described by 3 measuring functions, and again for each function 10 different attributes are associated to them. Totally, we can choose among 30 different diagrams, and we show here just the best result, which is the diagram with the biggest bounded area.

The second database (*Data2*) is composed by 80 scenes divided in 4 classes, where, respectively, there are a chair, a table and a cup in the first, a chair, a table and a human in the second, a chair, a table and a teddy bear in the third, and, in the last, a table, a human and a cup. Again, the best precision-recall diagrams are presented in Figure 4.27.

The diagrams show that the method performs similarly on the two datasets. Moreover, we can observe that in most of the cases the height function used for extracting graphs is the most performing, because all the models are oriented (in the Euclidean space) in the same way; in fact in our experiments we try to deal with scenes from the real world, and consequently with the same orientation. Finally, it is worth noticing that, in *Data2*, the method performs better on the class of scenes with a chair, a table and a human (see Figure 4.27(b)) than the ones with a chair, a table and a cup (see Figure 4.27(d)): this is due to the fact that human graphs are richer of features (articulations) than cup ones, and consequently the shape graph of the whole scene is more characteristic, simplifying their identification.

## 4.2 Shape Classification

We formalize a set of classification rules relying on a dissimilarity-based approach [PDP06], that is, we build classifiers defined in the dissimilarity measurement space, rather than in

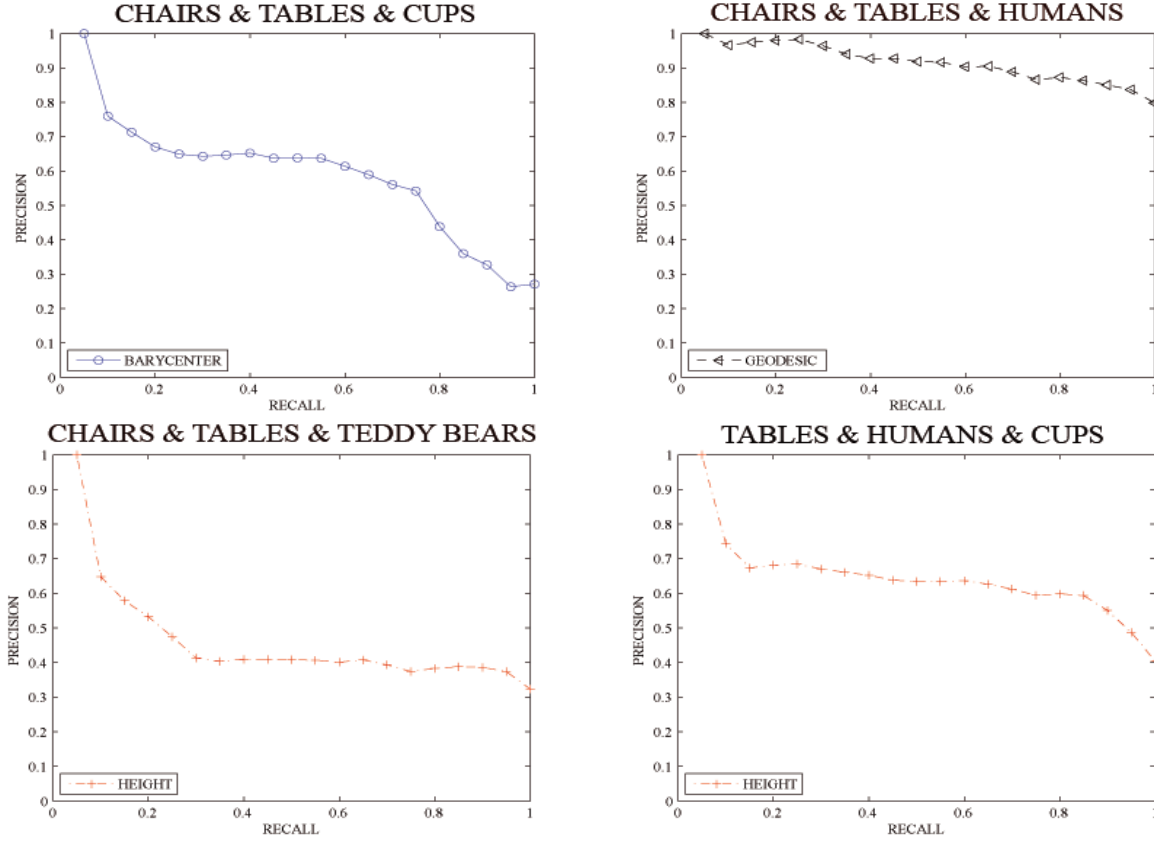


**Figure 4.26:** Precision-recall diagrams on the four classes of Data1

the space of feature descriptors. We consider the 1-NN rule and the use of both selective and creative prototypes.

The framework presented in this Section is general and can be applied to every kind of descriptor, and the proposed rules are very simple and intuitive. More powerful state-of-the-art classifiers (SVM, AdaBoost, ...) could have been taken into account, but they are not included in our study, since our focus is on the evaluation of the shape prototypes effectiveness. Indeed, the use of simple rules allows to better separate out the influence of the type of classifier from the actual properties of prototypes, that we want to investigate on.

Suppose we are given a database  $D$  containing  $n$  models, which are grouped into  $m$  classes of *similar* objects. More formally, given a database  $D = \{M_i\}$ ,  $i = 1, \dots, n$ , the  $n$  models are grouped into  $m$  classes  $C_k$ ,  $k = 1, \dots, m$ , so that  $C_k \neq \phi \forall k$ ,  $\bigcup_k C_k = D$  and  $C_k \cap C_l = \phi$  for  $1 \leq k, l \leq m, k \neq l$ .



**Figure 4.27:** Precision-recall diagrams on the four classes of Data2

We compute for each model  $M_i$  its descriptor  $S_i$  and consider a dissimilarity measure  $d$  between descriptors. The distance  $d$  is used to derive a query-to-class *membership measure*  $\tilde{d}$ , so that we classify a query  $Q$  by selecting the class  $C_{\bar{k}}$  which minimizes the membership measure  $\tilde{d}$  between the query and the class. In symbols:

$$Q \mapsto C_{\bar{k}} \iff C_{\bar{k}} = \arg \min_{C_k \in D} \tilde{d}(Q, C_k).$$

We consider a dissimilarity representation of the database  $D$ , that is an  $n \times n$  dissimilarity matrix  $\Delta_D$ , where each entry  $\Delta_D(i, j) = d(S_i, S_j)$  corresponds to the dissimilarity measure between the shape signatures of the objects  $M_i$  and  $M_j$ . In addition, we represent a query model  $Q$  by a vector  $\delta_D(Q) = (q_1, \dots, q_n)$ ,  $q_i = d(S_Q, S_i)$ , where the vector components are defined by the dissimilarity measurement between the query and the other objects in the database. The classification of  $Q$  is then performed by invoking classifiers living in the dissimilarity space, and operating on the dissimilarity vector  $\delta_D(Q)$ .

We analyze two classification schemes. The first classifier we consider is based on the Nearest

Neighbor rule. Let  $N$  be the set of indices  $N = \{1, \dots, n\}$ , and let  $N_k, N_k \subset N$ , be the set of indices corresponding to the models in the class  $C_k$ ,  $k = 1, \dots, m$ . The distance  $\tilde{d}(Q, C_k)$  is defined as the minimum distance between the query descriptor  $S_Q$  and the descriptors belonging to  $C_k$ :

$$\tilde{d}(Q, C_k) = \min_{i \in N_k} d(S_Q, S_i) = \min_{i \in N_k} q_i. \quad (4.5)$$

The second classification rule we consider takes advantage of the use of *shape prototypes*, to be used as representatives of the entire database. In other words, we define a set  $R$  of  $t$  shape prototypes  $\{P_i\}$ ,  $i = 1, \dots, t$ . The cardinality of  $R$  satisfies  $t \ll n$ , with  $n$  the size of the entire database. Considering that the prototype extraction can be performed off-line, this allows to reduce the number of dissimilarity measures to be computed at run-time.

When considering a single prototype  $P_k$  for each class  $C_k$ , the query  $Q$  is classified by selecting the class  $C_k$  which minimizes the membership measure  $\tilde{d}$  between the query and the class, where  $\tilde{d}(Q, C_k)$  is defined as the distance between the query descriptor  $S_Q$  and the prototype representing  $C_k$ . In symbols:

$$\tilde{d}(Q, C_k) = d(S_Q, P_k). \quad (4.6)$$

If  $t > 1$  prototypes are considered, we can apply a formula similar to Equation 4.5:

$$\tilde{d}(Q, C_k) = \min_{i \in T_k} d(S_Q, P_i) \quad (4.7)$$

with  $T_k$ ,  $|T_k| = t$ , the set of indices corresponding to the  $t$  prototypes representing the class  $C_k$ .

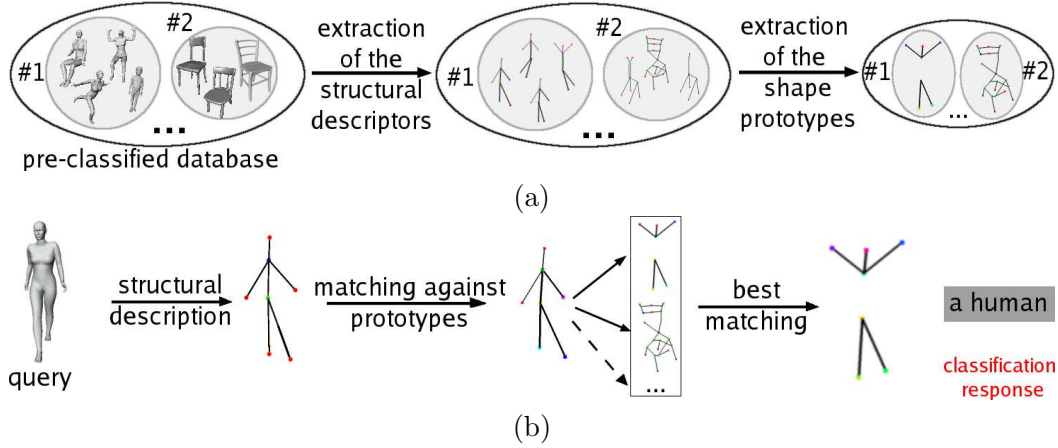
Prototypes, also known as representatives or templates, can be easily embedded in the classification framework. We distinguish between *selective* and *creative* ones: in the first case, one or a set of few members of the class are chosen as the best representatives of the whole class; in the second case, new models, that may not correspond to any member of the class, are generated as prototypes of the class.

Selective representatives are somewhat linked to the classification rules adopted. For example, a criterion is to choose as *selective prototype* the model  $P_k$  for a class  $C_k$  whose dissimilarities values are closer to the average ones.

Informally speaking, we consider a model as a class representative if it is *not too distant* from the other objects in the same class, or, in other words, if it is a sufficiently *inner* model with respect to its class. More formally, for each descriptor  $S \in C_k$ , we compute the following value, that we name *eccentricity*:

$$avg_{C_k}(M) = \frac{\sum_{R \in C_k} d(S, S_R)}{|N_k|} \quad (4.8)$$

with  $|N_k|$  the number of elements in the class  $C_k$ . Then, we may want to choose as selective prototypes of a class  $C_k$  those maximizing  $avg_{C_k}(\cdot)$ , that is to say *inner* models. Also,



**Figure 4.28:** Overview of the classification process when class prototypes are considered.

so-called *boundary* representatives, that is those maximizing the eccentricity, are interesting representatives of the class and they can be used with the average ones in order to provide a more complete summarization of the shape variability within the class (see Section 4.2.2). Finally, note that selective prototypes can be obtained directly from the dissimilarity matrix without any interaction with the shape descriptor. One possible methodology for defining creative prototypes will be presented in the next Section.

The flow of 3D shape classification using shape prototypes is illustrated in Figure 4.28. For each class, a small set of *prototypes* is defined: in the example, these are prototypes based on structural descriptors (Figure 4.28(a)). This phase works off-line and consists of a training over a pre-classified dataset.

The classification of an unknown object is then done by reasoning only on these subset of models, thus reducing largely the computational complexity of the classification. The classification of a new query is done at runtime matching the shape descriptor of the query against the class prototypes, see Figure 4.28(b). Once the shape prototypes, either selective or creative, have been extracted for each class  $C_k$ , the on-line phase of the classification pipeline is done using the rules in Equations 4.6 or 4.7.

#### 4.2.1 Creative prototypes using structural descriptors

The adoption of creative prototypes allows for more flexibility in the selection of the features to be included in the templates. The aim of *creative prototypes* is indeed to summarize in a new descriptor the relevant shape features of the members belonging to the same class. The goal is to identify the underlying information shared by the descriptors representing the objects in a class, and to be able to code at some extent the semantics needed for classification



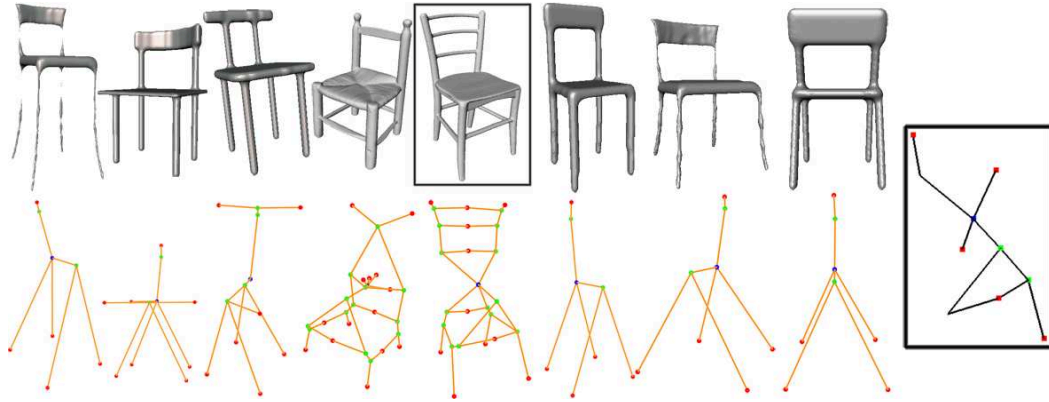
purposes. Structural descriptors are particularly useful to reach this goal, as they provide a homogeneous and explicit representation of relevant shape features and their adjacency-based configuration. Structural descriptors are frequently coded using graphs, where nodes store information about the main parts of the shape and the arcs store the adjacency, or configuration, of these features. Attributes are also used to augment the description with more geometric information. Several examples of such descriptors can be found in the literature, some of which presented in applications to 3D retrieval [TS05, HSKK01] and some for different applications [KT03, LV99, BMMP03], but in principle usable for classification as well.

Given two structural descriptors coded as graphs, it is possible to use graph matching techniques to effectively measure their similarity. More precisely, the identification of both common and different sub-parts, or sub-graphs, can be effectively used to somehow quantify and specify the reasons for the computed similarities. An example of graph-editing distance was presented in [BMSF06], which can be applied to directed a-cyclic and attributed graphs and is based on the identification of the common sub-graphs.

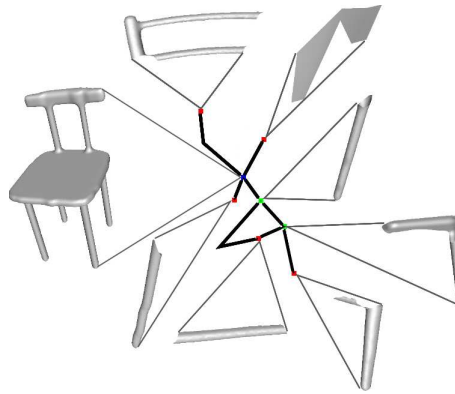
Structural descriptors and graph-editing distances can be coupled to provide a flexible strategy for defining creative prototypes. In the following, we briefly outline the specific method discussed in the evaluation Section, while we refer to [MSF07] for a detailed description of the method and a comparison with related methods [JA98, NB05]. The methodology for creating shape prototypes, extracted by structural descriptions of the shape, is made of two steps. First, a seed prototype is selected among the shape descriptors of a given class, and it is matched against the remaining descriptors in the class, using the common sub-graph method presented in [BMSF06]. For each comparison, the editing operations - i.e., node or arc deletion and addition, and attribute modification - that characterize the matching procedure are stored, and they will play a key role in the class prototype extraction. In the second step, indeed, a subset of these editing operations are selected and applied to the seed prototype to transform it into a new graph-based descriptor, which captures the relevant features shared by the members of the class. Notice that the resulting prototype are still directed a-cyclic and attributed graphs, with possibly many connected components. Concerning the selection of the seed prototype, different heuristics can be used, like the one with minimal average distance or the one with maximal eccentricity.

Obviously, the expressiveness of the shape prototype depends also on the expressiveness of the structural descriptor. In our case, this is the *Shape graph (SG)* enriched with region-based geometric attributes described in Section 3.1.1.2 and [BMSF06]. As function  $f$  to drive the SG extraction we adopt the geodesic function [HSKK01] that detects protrusions and performs well for matching articulated objects in a pose-independent manner.

In Figure 4.29 we show a set of chair models (upper row) with the corresponding SG descriptors (lower row). The prototype generated from the sequence of editing operations is shown on the right hand side of the Figure. Note that the prototype is still a directed,



**Figure 4.29:** A set of models (upper row) and their SG descriptor (lower row) and the class prototype (right).



**Figure 4.30:** Details on the geometric attributes of the prototype obtained in figure 4.29.

acyclic attributed graph, that summarizes at an abstract level the main characteristics of the models in the class.

Figure 4.30 shows the attributes, that is the shape parts, associated to the nodes of the structural prototype. This highlights that the new descriptor is a composition of the elements that are the most characteristic of the class; indeed, we can notice for instance the rear part of a chair and its legs.

### 4.2.2 Evaluation

3D shape classification has been run on the benchmark of 400 models used for the Watertight Track of SHREC (SHape REtrieval Contest) 2007 [GBP07] shown in Figure 4.17. To the benchmark of 400 models, we have added a set of 80 models (Figure 4.31) to be used as



**Figure 4.31:** 80 models used as queries when evaluating the classification framework.

external queries against the 400 elements of the benchmark. Notice that the 400 objects also serve to compute the representative models, both selective and creative.

The first performance indicator we compute is the *classification rate*, that is to say the percentage of query models which are correctly classified. Secondly, since a membership measure  $\tilde{d}$  provides a ranking for the set of classes, we evaluate the rank at which the right class of a query is recognized: the smaller the score, out of 20 (i.e. the number of classes in the database), the better the performance. The results are reported in Table 4.4, that includes the classification rules formalized in Section 4.2. Position is the average position of the correct class with respect to the ranking identified by the classifier; for the sake of completeness, its floating point value is reported, although it should obviously be transformed into an integer value. A selective prototype is chosen according to the eccentricity parameter defined in Equation 4.8, and a creative prototype is built following the approach in Section 4.2.1. The results are averaged over the 80 external queries matched against the 400 models of the database.

	1-NN	Selective Prototype	Creative Prototype
<b>Classification Rate</b>	85%	61%	66%
<b>Position</b>	1.4	2.5	2.2

**Table 4.4:** Classification rate indicates the percentage of query models which are correctly classified.

The highest values for the classification rate are reached by using the Nearest Neighbor Rule. This classification scheme allows to obtain a perfect score by considering only the first 2 classes in the system response.

The classification schemes based on the selection or creation of a prototype model show lower performances, but still convincing, if we consider that the added value of these classification approaches is that they allow a strong reduction of the number of comparisons to be performed at run time. An alternative solution to the problem of dimensionality reduction could be the use of **indexing** techniques.

The fact that the creative prototype improves the results of the selective members (of about

8%) indicates that creative prototypes are more suitable than selective ones to exploit the information contained in a class, encoding a number of distinctive properties shared by the class they represent. It is possible, indeed, that the information characterizing a class of models is not entirely collected in a single model or a small subset of models; on the other side, prototypes are built taking into account all the objects in the class, and are able to code in a compact way this amount of information.

In order to increase our confidence in this result, we decided to compare the performance of selective and creative prototypes by varying the size of the training database and the number of queries. This is indeed an important issue in the evaluation of the classification performance. In particular, the ratio between the size of the training set and the number of queries evaluated was put to 380 queries over 100 training elements, 280 over 200, 180 over 300 and 80 over 400. We also evaluated the use of a number greater than one of prototypes for each class.

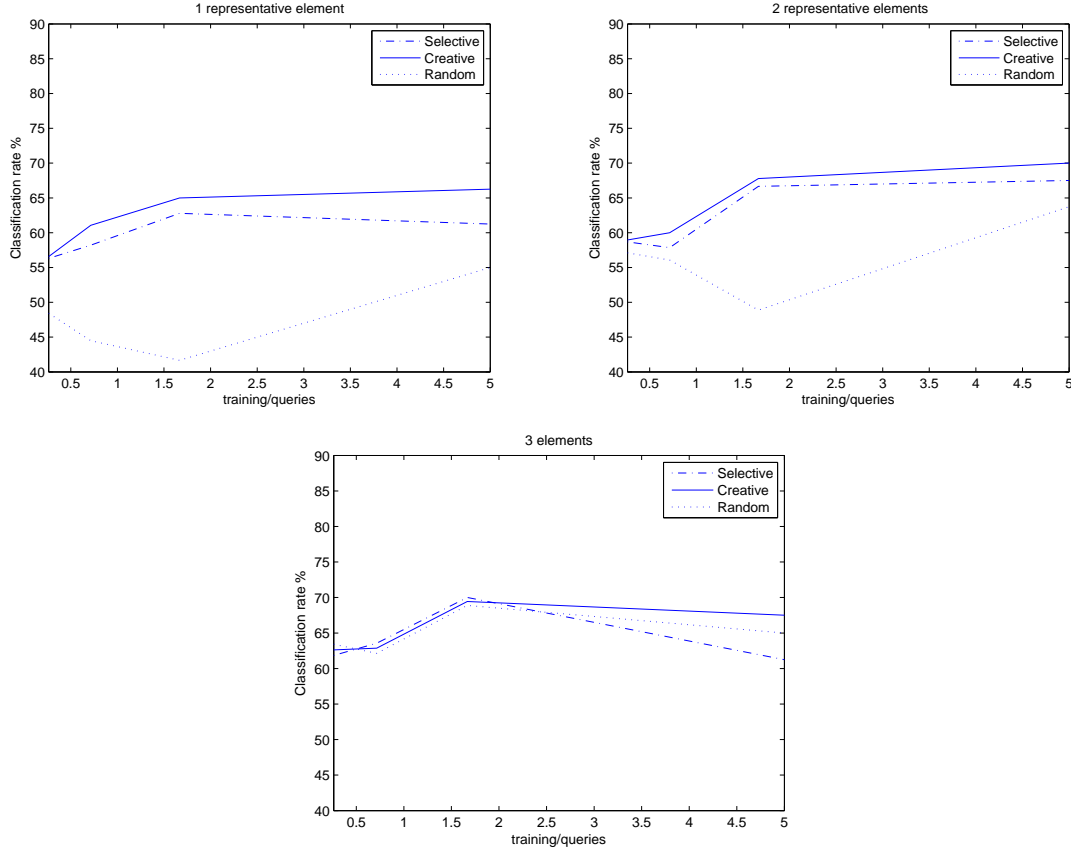
Figure 4.32 compares the classification rate obtained by using creative prototypes with the results obtained by selective prototypes, and also by a set of randomly chosen class prototypes. The abscissa in the three plots represents the ratio between the size of the training set and the number of queries evaluated; the starting value corresponds to the experiment with 380 queries and 100 training elements, while the final value corresponds to the situation with 80 queries and a training set of 400 models. The plots in Figure 4.32, from top to bottom, refer to the performance obtained using 1, 2 or 3 representatives per class, respectively. In particular, in the first and the second case both the selective and creative prototypes are built selecting the models showing lower eccentricity values (1 model in the first case, 2 models in the second one). When 3 representatives are selected, we consider the less, the more and the average eccentrical models. The latter choice is the result of a series of experiments, not included for space reasons.

The first three graphs show that, when a small number of representatives per class is used, the performance using the set of creative prototypes significantly improves the classification performance with respect to selective and random prototypes, confirming the conclusion derived from Table 4.4. It makes sense to compare the performance of the sets of selective and creative ones, since their members are selected according to the same criterion related to eccentricity. The fact that the set of randomly chosen selective models provides the worst performance is a confirm that the eccentricity is a reasonable parameter to select class representatives.

According to our results, a small number of prototypes is able to encode class information, better than selective prototypes do. We consider this result really promising in the direction of dimensionality reduction and efficiency improvement. The experiments in the following Section will show the capability of shape prototypes to support fast and reliable shape retrieval.

We discuss how a good classification scheme is able to improve a shape retrieval system:

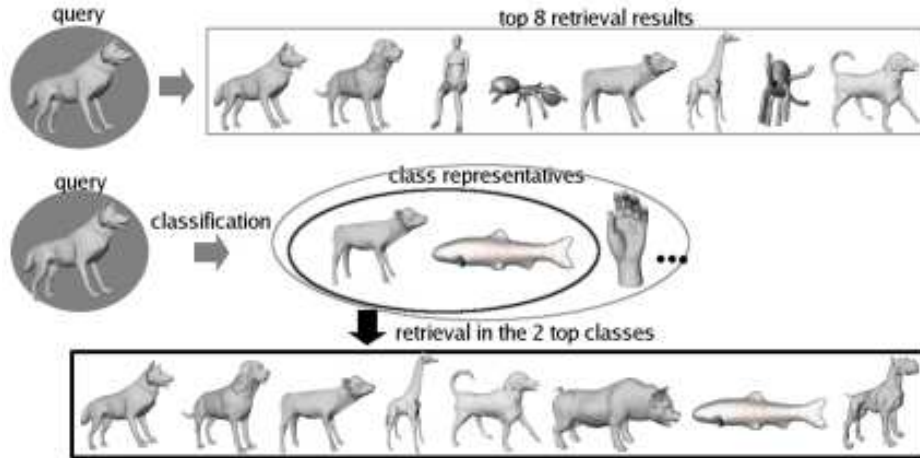
---



**Figure 4.32:** Comparison among the classification rates using one element (top), two and three elements (resp. middle and bottom) per class from selectively and randomly chosen representants and our Creative prototypes. The abscissa is the ratio between the size of the training set and the number of queries.

indeed, if classification is involved as a preliminary step in the retrieval pipeline, it reveals to be useful to minimize both the number of internal comparisons and the number of false positive answers.

The retrieval pipeline based on prior classification can be easily illustrated through the example in Figure 4.33. Performing retrieval in the classical way yields the results in 4.33(top), where the top 8 retrieval results for a wolf model are shown with respect to the whole database. If the query is beforehand classified, the search can be restricted to the top classes in the classification rank: in this example, we select the first 2 classes returned by the system (Figure 4.33(middle)), thus considering 40 models instead of 400, i.e. 10% of the original data. Moreover, Figure 4.33(bottom) shows that prior classification of the query improves the results of the retrieval process in terms of precision, by discarding some of false positives. We recall that by false positives we mean items which are judged to be similar to a given



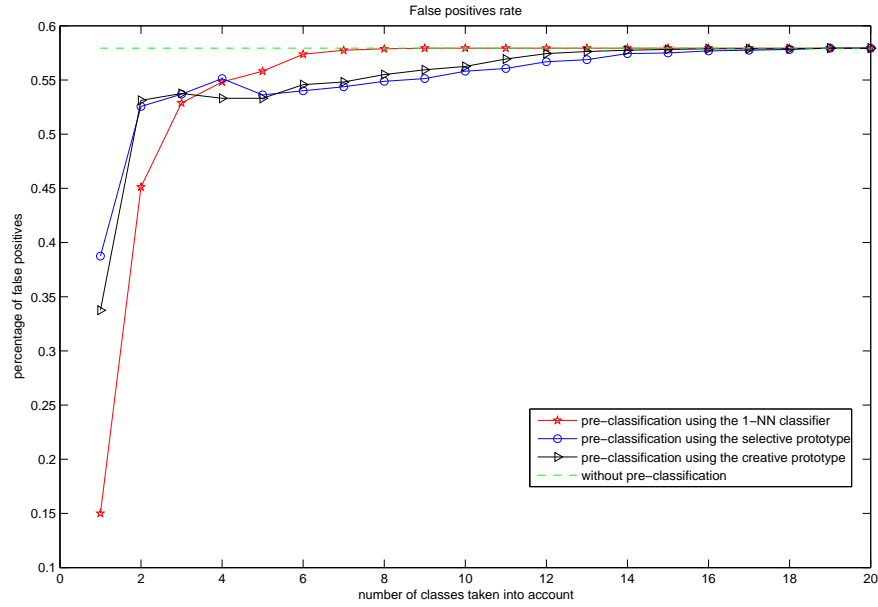
**Figure 4.33:** Improvement of a query answer when pre-classification is involved in the retrieval pipeline.

query (i.e., that are retrieved within the first items) although they do not represent correct matches.

The diagram in Figure 4.34 analyzes the improvement of the precision when pre-classification is involved in the retrieval process. The plotted lines represent the average percentage of false positives within the first 20 ranked items, plotted against the number of classes taken into account. More precisely, when a query model is submitted for retrieval, the classes of the database are ranked according to a given membership measure. An integer  $p$  is chosen,  $1 \leq p \leq 20$ , and only the models belonging to classes up to the  $p$ th-ranked one are allowed as possible answers to the query. The percentage of false positives over the first 20 retrieved members is computed and plotted versus the varying number  $p$ . The three curves refer to the different classification methods proposed, namely the 1-NN rule and classifiers based on selective and creative prototyping.

Figure 4.34 clearly shows how the number of false positives is strongly reduced when pre-classification is applied and a small number  $p$  of classes is considered. Without pre-classification, the computed percentage of false positives using SG is 58%. Performing 3D shape classification allows to obtain a strong reduction of wrong results: in our experiments, the percentage of false positives is reduced to be 15%, 38.75% and 33.75%, respectively using the 1-NN rule or choosing a single selective or creative prototype per class. Once again, we observe that creative prototypes perform better than selective ones, while it is not surprising the overall good performance of the 1-NN.

It is also interesting to observe the non-linearity of the growth of the plotted curves. Even if the low values at the very beginning of the plot confirm that a few classes are enough



**Figure 4.34:** False positive retrieval results, averaged over 80 queries against 400 models. Value  $0.y$  means  $y\%$  percentage of false positives, and is plotted vs the number of classes taken into account after pre-classification.

to obtain good retrieval results, the behavior of the curve shows that the precision of the retrieval improves if only the top ranked classes with respect to the query are considered. Considering a larger number of classes – i.e. enlarging the search to consider up to a 30% of data models – no longer improves results.

### 4.3 Best view selection

In this Section, we focus on the quantification and measurement of the visual information present in an image of a 3D object with the aim of finding optimal, or nearly-optimal, views. It should be emphasized that the notion of the *goodness* of a view may depend on the particular visual task or application. For example, in an illustrated manual of work tools, people may prefer views where the tool is drawn in the typical position, as used by the machine operator. Object recognition tasks performed by a robot may require a totally different view to achieve best performance. Nonetheless, we believe that there exists some common basis for all these visual problems.

Answering these questions presents a significant challenge in the field of visualization and shape understanding. A solution would be useful in several applications such as automatic

camera positioning in CAD, thumbnail generation for large 3D databases, automatic scene composition, technical illustration, and object recognition.

In this Section we propose the following methodology: define a view *descriptor* which attaches a score to a view of the object, taking into account its visible geometry (Section 4.3.1). Then, compute the value of this descriptor for a small number of candidate views (Section 4.3.2). We consider the view with the highest score to be the most informative. We describe a number of such descriptors, and show how to optimize them efficiently over the viewing sphere. We compare the views generated by these descriptors and discuss their performance (Section 4.3.3).

### 4.3.1 View descriptors

In this section, we describe a number of ways to measure the goodness of a view of an object. The objective function that measures this is called a *view descriptor*, and the best view is that which maximizes this function. Our descriptors are based on the following principles.

The first principle is to exploit an accepted measure of geometric complexity for a 3D shape. This could be based on various features in the shape, its surface area, its curvature distribution, etc, and is obviously view-independent. The view descriptor would then assign to a view a score which is the contribution to the complexity from the portion of the shape which is *visible* in that view (see Sections on *Surface area entropy*, Section 4.3.1.1, *Visibility ratio* (Section 4.3.1.2) and *Curvature entropy*, Section 4.3.1.3). In our understanding, we define as the best view the one that exposes as much of the geometric complexity of the object as possible.

The second principle is to define descriptors which are based on inherently view-dependent features [ABM<sup>+</sup>06]. Examples are object silhouettes and critical points (see Sections *Silhouette entropy* (Section 4.3.1.5) and *Topological complexity*, Section 4.3.1.6). Here again we would like to expose as much of these features as possible.

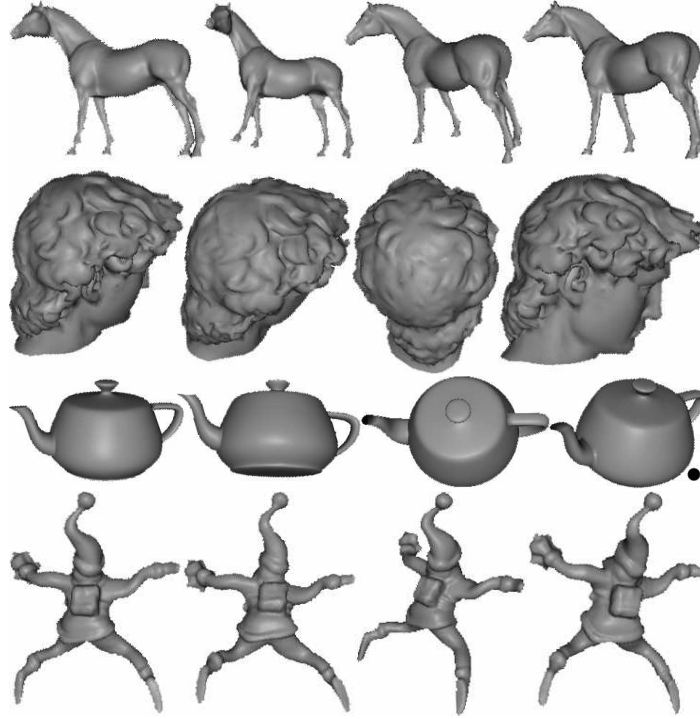
A third principle is to build a descriptor which, instead of assigning values to the primitive elements of the 3D model (e.g., vertices, faces, and edges), assigns values to larger portions of the model which have some semantic meaning. Such portions of the model may be obtained from segmentation algorithms (see Section 4.3.1.7 *Surface entropy of semantic parts*). This affords a higher level visual appreciation of the model.

#### 4.3.1.1 Surface area entropy

The first descriptor that we examined measures geometric complexity of an object as its surface area. Each face is assigned a “probability” — the fraction of its visible projected area relative to the total visible projected area and the descriptor value is the entropy of this

---





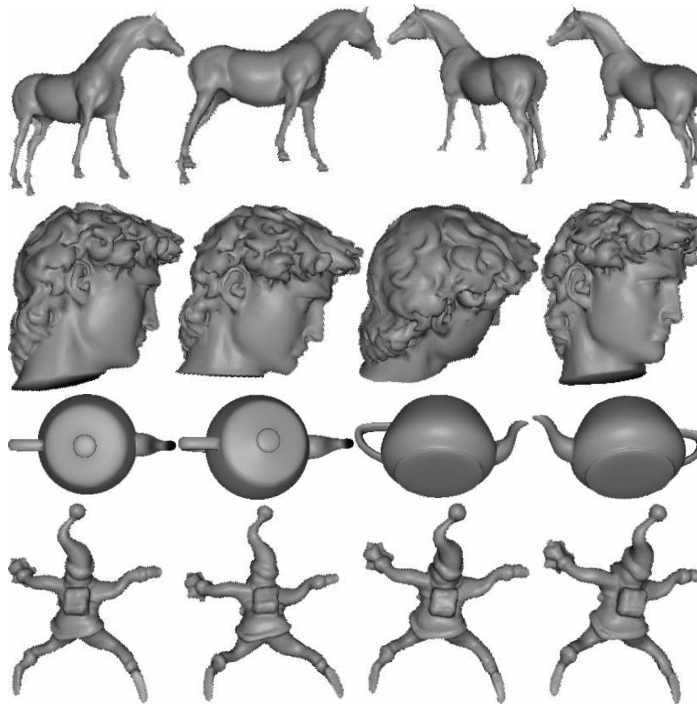
**Figure 4.35:** Four top-ranking (left to right) views among candidate views according to the surface area entropy descriptor. Black dots indicate a three-quarter view (otherwise it is a normal clustered view).

distribution. We computed these probabilities at image precision by rendering each face with a distinct colour, and counting the number of pixels of each colour. This is essentially the “viewpoint entropy” method proposed by Vázquez et al. [VFSH01]. The ranking of some views by this descriptor are shown in Figure 4.37. In this figure, and those related to the other descriptors, we restrict our attention to a small number of candidate views generated by a *filtering* procedure, as described in Section 4.35.

#### 4.3.1.2 Visibility ratio

A shape descriptor based on surface entropy did not take into account the behavior of the *invisible* portions of the surface, so it might prefer a view of the object in which most of its surface area is occluded. A descriptor which does take this into account is the ratio between the 3D surface area that is visible in the image, and the total 3D surface area. This seeks to expose as much of the surface area as possible. The ranking of some views by this descriptor are shown in Figure 4.36

.



**Figure 4.36:** Four top-ranking (left to right) views among candidate views according to the visibility ratio descriptor.

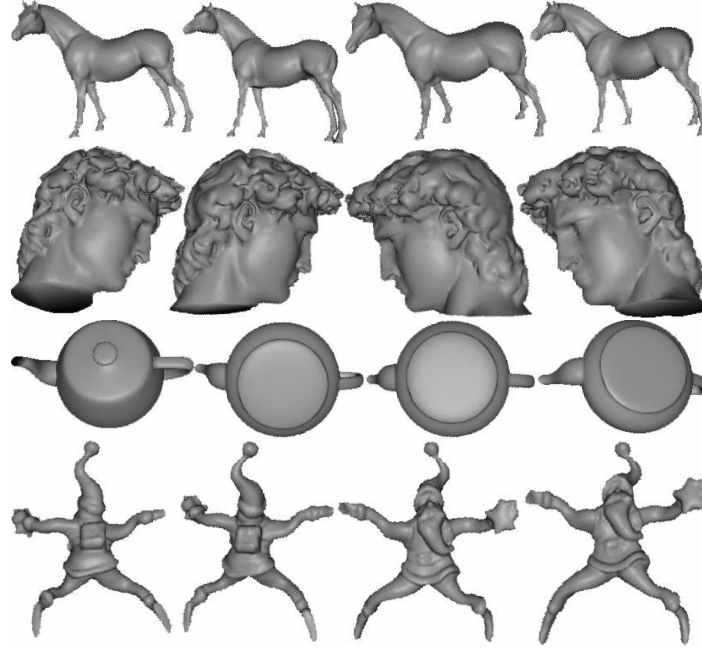
#### 4.3.1.3 Curvature entropy.

Surface area is a very simple measure of shape complexity. A more sophisticated one, as proposed by Page et al [PKS<sup>+</sup>03], is the entropy of the Gaussian curvature distribution over the entire surface of the object. We define the curvature entropy descriptor to be the entropy of the curvature distribution over the *visible* portion of the surface. The curvature at a vertex  $v$  is estimated by the standard angle-deficit approximation, as in [PKS<sup>+</sup>03]. The ranking of some views by this descriptor are shown in Figure 4.37.

The previous descriptors were based on view independent measures of shape complexity. We now define descriptors which are inherently view-dependent.

#### 4.3.1.4 Silhouette length

Silhouettes (sometime called “occluding contours”) seem to provide an accurate and compact depiction of the shape of a 3D model, and, for this reason, are often used in non-photorealistic rendering. Silhouettes are also view-dependent. A simple version of this descriptor measures the total length of all silhouette edges in the image plane. Since this cannot be done reliably in image space, we computed the visible silhouette edges in object space analytically and



**Figure 4.37:** Four top-ranking (left to right) views among candidate views according to the curvature entropy descriptor. Black dots indicate a three-quarter view (otherwise it is a normal-clustered view).

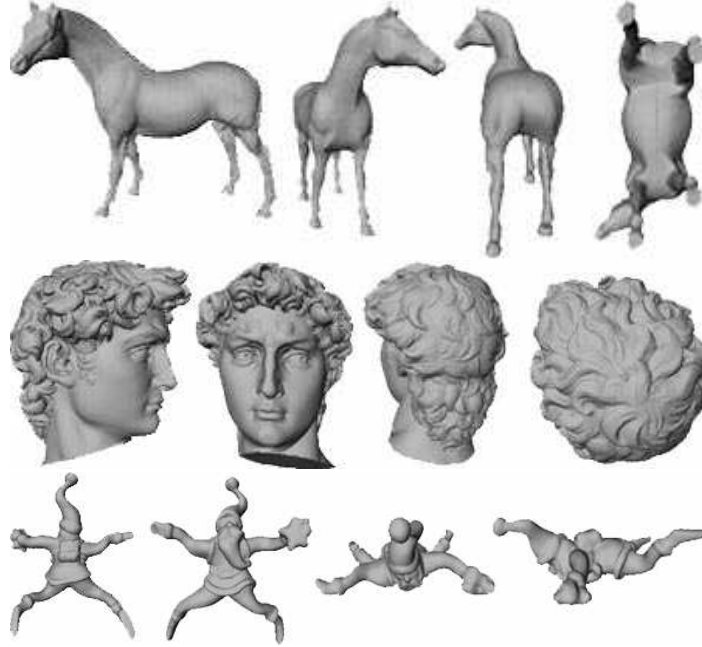
calculated the length of their projected versions. The ranking of some views by this descriptor are shown in Figure 4.38.

#### 4.3.1.5 Silhouette entropy

A more sophisticated silhouette-based descriptor uses silhouette entropy instead of total length, where the entropy of a curve is defined as the entropy of its curvature distribution, as proposed by Page et al [PKS<sup>+</sup>03]. In the discrete version, we compute the entropy of all turning angles between adjacent silhouette edges. In some cases, spurious silhouette edge crossings can make the result quite unstable. The ranking of some views by this descriptor are shown in Figure 4.39.

#### 4.3.1.6 Topological complexity

Another approach has been originated by the shape characterization approach adopted to build the Shape Graph, see Section 3.1. Its relevance is due to the fact that the critical points of a 3D surface are highly informative. So critical points, and more in general the critical regions that correspond to the SG nodes, may be considered as *salient features* of the surface. From the theoretical point of view, if the surface is a smooth, closed surface  $D$

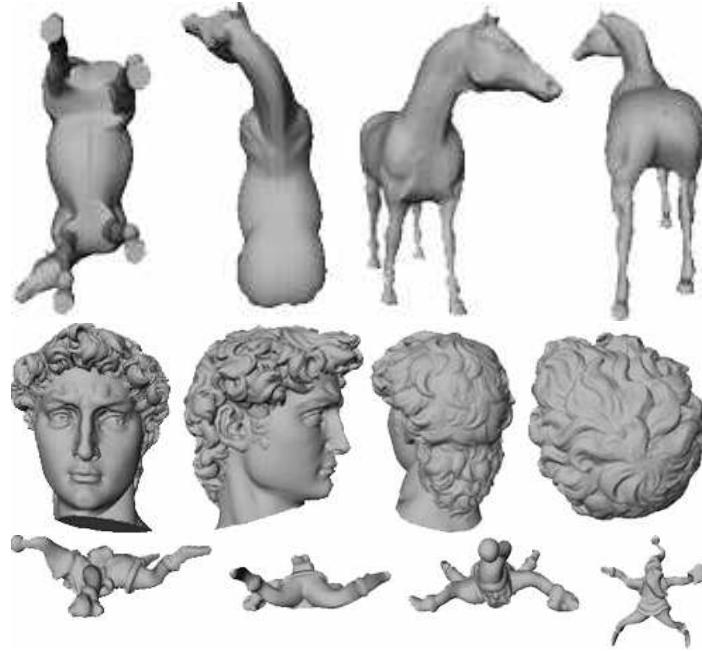


**Figure 4.38:** Four top-ranking (left to right) views among candidate views according to the silhouette length descriptor. Black dots indicate a three-quarter view (otherwise it is a normal-clustered view).

embedded in  $\mathbb{R}^3$ , we may use the height function  $h_n$  along any direction  $n$  and compute the number of its critical points (i.e., minima, maxima, and saddles) on  $D$ . A classical theorem from differential topology [Mil63] states that the alternate sum of the number of minima, maxima, and saddles is constant, and is related to the Euler characteristic  $\chi$  of  $D$ , namely:

$$\text{maxima} - \text{saddles} + \text{minima} = 2(1 - g) = \chi, \quad (4.9)$$

where  $g$  is the genus of  $D$ . In particular, the result (4.9) is valid not only for Morse functions on the surface, but also for piecewise-linear functions [Ban70]. In the computational settings of our implementation, critical points are replaced by critical areas. As shown in Section 3.1, under the hypothesis that the contouring approach described in [Bia05] is sufficiently dense, also critical areas verify an extended version of Equation 4.9. However the total number of critical points (maxima+saddles+minima) depends on the direction  $n$ , this quantity could be useful for discriminating among different view directions (when used as  $n$ ). The direction that maximizes this number seems to be the most informative. Therefore, it can be applied even if the height function has degenerate critical points. The ranking of some views by this descriptor are shown in Figure 4.40.



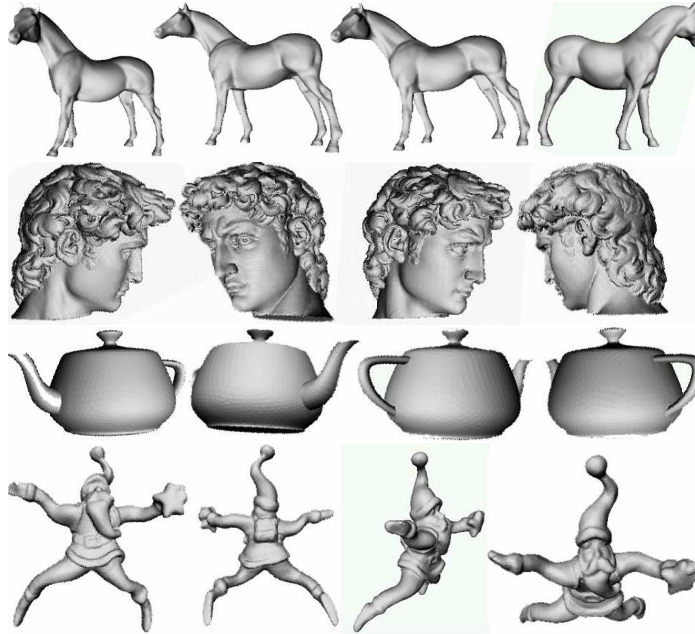
**Figure 4.39:** Four top-ranking (left to right) views among candidate views according to the silhouette entropy descriptor. Black dots indicate a three-quarter view (otherwise it is a normal-clustered view).

#### 4.3.1.7 Surface entropy of semantic parts

It is possible to apply the surface area entropy method to geometric elements larger than the primitive elements (e.g. vertices, faces) of a 3D mesh. One way to achieve this is to use semantically important segments of the model. The probability of each segment is defined to be the visible projected area of this segment relative to the visible projected area of the entire model. There exist many mesh segmentation algorithms and the descriptor will depend critically on the segmentation method. In our experiments, we used the method proposed by Dey et al. [DGG03], which seems to be able to identify parts of the model which are semantically meaningful (e.g. nose, ears, neck, etc for a head model). The ranking of some views by this descriptor are shown in Figure 4.41.

#### 4.3.2 Sampling the view space

Given a view descriptor measuring the “goodness” of a view as a function of viewing direction, the problem is then to find the global, or even local, maximum of this function over the viewing sphere. Since the search space is a continuum containing an infinite number of points, we have used two different strategies to reduce the search to an exhaustive search on a small but reasonable set of candidate views. In this section, we describe two methods to



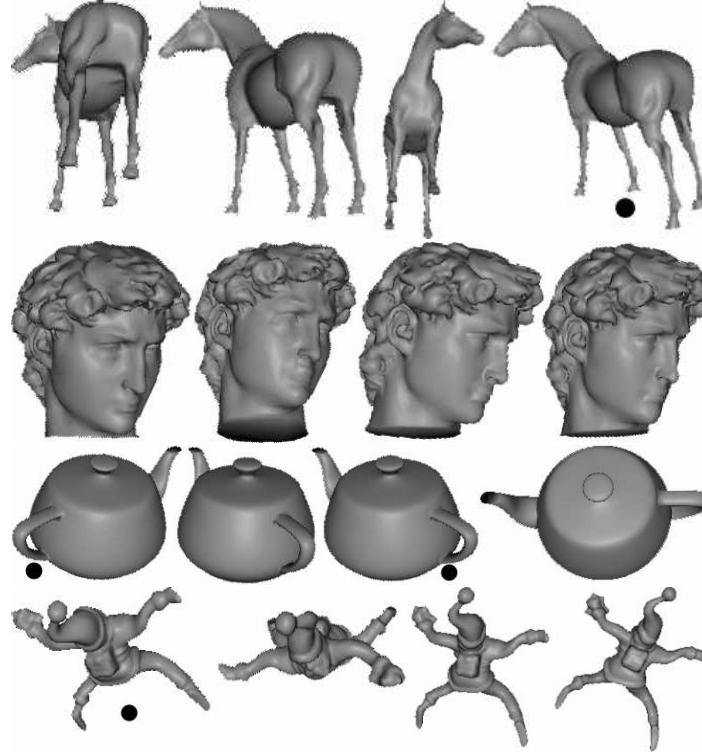
**Figure 4.40:** Four top-ranking (left to right) views among candidate views according to the topological complexity descriptor. Black dots indicate a three-quarter view (otherwise it is a normal-clustered view).

generate these candidate views.

For many inputs, nature dictates an “up” direction, which should be respected in any view, certainly in the best one. So, for example, an animal should not be rendered upside down, rather standing on its feet. Noticing that the best view has a degree of freedom of 2D rotation in the image plane (since this will not change the value of any of the descriptors), it is possible to exploit this degree of freedom to cause all views to have the correct 2D orientation after the optimal view has been computed.

#### 4.3.2.1 Three-quarter views

Palmer et al. [PRC81] experimentally observed the existence of what they call “canonical views” of an object. These are views that most humans prefer to look at the object from, and they seem to be quite well defined in practice. Inspired by the structural description theories, Blanz et al. [BTB99] state that these canonical views are typically “three-quarter views” of the objects. Gestalt psychologists explain that this is a view where the front, top, and side of the object are simultaneously visible. This also explains why these views are preferred by humans: we simply prefer to see simultaneously all three dimensions of the shape. In particular, Marr [Mar82] states that the object’s “primary axis of elongation” should be clearly visible.



**Figure 4.41:** Four top-ranking (left to right) views among candidate views according to the surface entropy of semantic parts descriptor. Black dots indicate a three-quarter view (otherwise it is a normal-clustered view).

We use three-quarter views as candidate views. We start by approximating the shape by an oriented box, thus establishing a local Cartesian frame defined by the three axes of the box. Three-quarter views then correspond to the vectors whose components are the eight combinations of  $(\pm 1, \pm 1, \pm 1)$  in this coordinate system. As pointed out by Weinshall and Werman [WW97], these views are the most stable views of the box approximating the object, thus hopefully a good approximation for stable views of the object itself.

To compute an approximating box for the object, we use the three principle directions generated by PCA of the object geometry, relative to the centroid of the point cloud. The PCA method has some nice properties, such as robustness and stability; furthermore, it has been successfully used in computer graphics for object matching and aligning and normalization purposes (e.g., see [ETA00, ETA01]). Finally, we note that the eight vectors are computed ignoring occlusion, which will be taken into account later by the descriptors.

### 4.3.2.2 Normal clustering

Another way to compute candidate views of an object is to detect clusters in the set of vertex normals. Again we ignore occlusions. This effectively defines the “sides” of the object to be those directions that a large number of normals point towards. This is motivated by the assumption that the more the normals point in some direction, the more the object’s surface is visible from that direction. More precisely, for each vertex  $v$  we approximate its unit normal by averaging the normal vectors of the triangles incident on  $v$ ; each normal vector defines a point on the unit sphere (i.e., the Gauss map). Then, we cluster the points on the unit sphere using an iterative version of PCA [Jol86], thus achieving a set of clusters. An interesting view is defined as the barycenter of these points (or equivalently, normal vectors) of each cluster. The resulting view directions seem to be quite stable.

### 4.3.3 Experimental Results and discussions

To compare the performance of the various descriptors, we applied them to a set of 3D objects which seem to be representative. For each object we computed a relatively small number of candidate views based on the two methods described in Section 4.3.2 and for each such view, computed the value of the various descriptors described in Section 4.3.1, and ranked them in decreasing order (from left to right), as depicted in Figures 4.35 – 4.41. The objective was to see whether those views which ranked highest according to some descriptors were indeed those which are most informative to a human observer.

The three-quarter view and normal clustering sample the view space by considering the geometry and normals respectively. The main difference between these two methods is the number of candidates generated. This is constant (i.e., eight) for the first method and essentially unbounded for the second method. In practice, we choose the 15 to 30 most significant clusters to emerge from the PCA. Three-quarter views are marked by a black dot in Figures 4.35 – 4.41. It seems that all the view descriptors prefer mostly the normal clustered views over the three-quarter views.

The problem of finding a good view for an object seems to be quite difficult. It is becoming painfully obvious that there is no panacea. No one descriptor does a perfect job. It is probably possible to improve the descriptors described here and fine-tune them a little more, but we do not believe that this will be significant. However, since each descriptor does a reasonably good job on a majority of inputs, we are confident that it is possible to combine them to amplify the advantage that each has. Possible combinations are linear, where the optimal weights will have to be determined by some learning process, or non-linear, e.g. by a voting process.

Once the descriptors have been decided on, an efficient algorithm must compute the view on the unit sphere which optimizes this measure. At first glance, this seems to be a difficult



problem, since there exists a continuum of possible viewpoints. A gradient-descent optimization over the viewing sphere could work, but it would be very slow and not guarantee a global maximum. Hence we reduce the problem to a search over a (possibly large but) finite set of candidate viewpoints but, consequently, we might miss the best view. It would be useful to be able to prove that the particular measure we use can be maximal only at the candidate views. As we have shown, possible candidates are three-quarter views or normals, but it is not obvious that the optimal view must indeed be one of these.

## 4.4 Discussions

The description framework discussed in Chapter 3 offers an effective solution to the problem of searching and retrieval. In particular, the shape descriptor detailed in Section 3.1.1.2 is suitable for sub-part correspondence. As shown in figures 4.16 and 4.15, this coupling is advantageous for comparing models having similar overall shape and structure, as well as only common sub-parts.

We have experimentally proven that structure-based matching can improve the retrieval performance in terms of both functionalities supported (i.e., partial and sub-part correspondences) and the variety of shape descriptions that can be used to tune the retrieval with respect to the context of application. From our experiments we have seen that the characteristics detected by the mapping functions contribute to localize important shape features as well. In fact, depending on the choice of  $f$ , the shape descriptors detect relevant morphological characteristics like protrusions, concavities, pockets, branchings, slots, narrowings and involutions, that users may combine with other information and vary according to their desiderata. Having selected the most appropriate mapping function, the associated descriptors are fully automatic and do not require any user interaction.

As future contribution in the direction of best view object selection, the work of Lee et al. [LVJ05] suggests to adopt a gradient-descent algorithm to optimize the visible *saliency* over the viewing sphere. In this way, it would be possible to define a multiscale curvature measure that could enhance the descriptors proposed in Section 4.3.1 to accommodate this type of information as well.

## Related publications

S. Biasotti, D. Giorgi, M. Spagnuolo, and B. Falcidieno. Computing size functions for 3D models. *Pattern Recognition*, 2008. in print.

A. Cerri, S. Biasotti, and D. Giorgi.  $k$ -dimensional size functions for shape description and comparison. In *ICIAP 2007: Proceedings of the 14<sup>th</sup> International Conference on Image*

---

*Analysis and Processing*, Modena”, September 10-14 2007. IEEE Computer Society Press.

S. Biasotti and S. Marini. 3D object comparison based on shape descriptors. *International Journal of Computer Applications in Technology*, 23(2/3/4):57–69, 2005.

S. Biasotti, M. Marini, M. Spagnuolo, and B. Falcidieno. Sub-part correspondence by structural descriptors of 3D shapes. *Computer Aided Design*, 38(9):1002–1019, September 2006.

S. Biasotti and S. Marini. Sub-part correspondence using structure and geometry. In *Proceedings 4th Eurographics Italian Chapter Conference*, pages 23–28, Catania, 2006. The Eurographics Association.

L. Paraboschi, S. Biasotti, and B. Falcidieno. 3D scene comparison using topological graphs. In *Proceedings 5th Eurographics Italian Chapter Conference*, pages 87–93, Trento, 2007. The Eurographics Association.

L. Paraboschi, S. Biasotti, and B. Falcidieno. Comparing sets of 3D digital shapes through topological structures. In F. Escolano and M. Vento, editors, *GbR2007: Proceedings of Graph-based Representations in Pattern Recognition*, volume 4538 of *Lecture Notes in Computer Science*, pages 114–125, Alicante, 2007. Springer Verlag.

S. Biasotti, D. Giorgi, S. Marini, M. Spagnuolo, and B. Falcidieno. A comparison framework for 3D classification methods. In *IW-MRCS '06*, LNCS, 4105, pages 314–321, 2006.

S. Biasotti, D. Giorgi, S. Marini, M. Spagnuolo, and B. Falcidieno. 3D classification via structural prototypes. In *Semantic and Digital Media Technologies*, LNCS, 4816, pages 140–143, 2007.

O. Polonsky, G. Patané, C. Gotsman, S. Biasotti, and M. Spagnuolo. What’s in an image? towards the computation of the ”best” view of an object. *The Visual Computer*, 21(8-10):840–847, 2005.

M. Attene, S. Biasotti, M. Mortara, G. Patané, M. Spagnuolo, and B. Falcidieno. Computational methods for understanding 3D shapes. *Computer & Graphics*, 30(3):323–333, 2006.

M. Attene, S. Biasotti, M. Mortara, G. Patané, M. Spagnuolo, and B. Falcidieno. Topological, Geometric and Structural Approaches to Enhance Shape Information. In *Proceedings 4th Eurographics Italian Chapter Conference*, pages 7–13, Catania, 2006. The Eurographics Association.

---

## Chapter 5

# Conclusions

The main theme of this dissertation is the study of a shape description framework suitable for shape recognition and matching purposes. The descriptors we have discussed move from a geometric model to a conceptual one, encoding the original shape in a concise description. The main challenge we have faced is to achieve a good balance between space storage and efficacy of the shape descriptor. In our approach, these skills have been addressed adopting descriptors that are flexible and may be tuned according to the user's needs. We have identified the shape descriptors based both on topology and geometry as the tools the most suitable for shape comparison and retrieval.

To this end, we have defined the Shape Graph representation and we have extended the effective computation of the size descriptor to surfaces and higher dimensional data, and we have shown their efficacy on a number of application domains.

In the following Sections, we sum up each of our contributions and we sketch future research directions.

### 5.1 Summary of results

The main contribution of this thesis is related to the computational aspects that support the definition of a conceptual model for 3D object representation based on abstract coding, and its effectiveness for shape recognition, matching and retrieval applications. The core of our technology is based on the development of algorithms that can robustly handle shapes while maintaining a formal framework.

The originality of this work is both theoretical, having defined a shape description framework able to deal with shape topology and geometry in discrete settings, and technological, having proposed methods for shape coding and retrieval. In particular, the formal framework behind the topics discussed in this thesis has been recognized as relevant from the CNR and founded

by a CNR grant: *Topology and Homology for analysing digital shapes*, DG.RSTL.050.004.

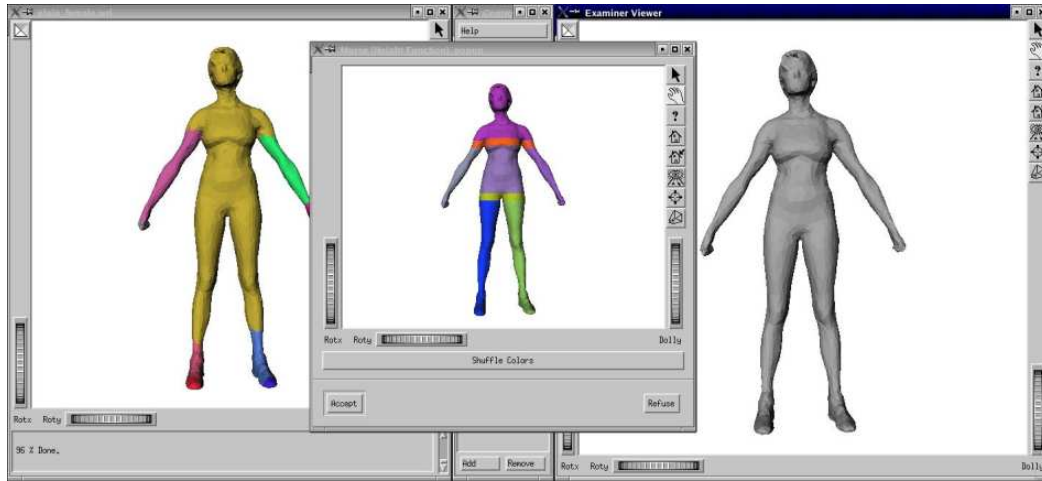
As a first contribution of the thesis, we have provided an original classification of the shape descriptors at the light of the properties of the real functions chosen for their definition and their flexibility with respect to this choice.

Furthermore, starting from a geometric model, mainly a triangle or a volume mesh, we have derived two descriptors, the Shape Graph and the size descriptor, which sketch the overall appearance of a surface shape and its topological type, being able to discard irrelevant details. Moreover, the structure of the Shape Graph is combined and enriched with geometric information, such as the position in the space of the points of the contour levels that bound the critical areas and the length of a protrusion, which may be used to better satisfy the application needs. In particular, we have suggested two ways to explicitly couple topology and geometry in a unique description and how to extend the Shape Graph representation to set of objects of models made of multiple parts. The relevance of the Shape Graph representation is in the independence of the computational framework on the choice of the real functions and the capability to capture both topological and geometrical shape information.

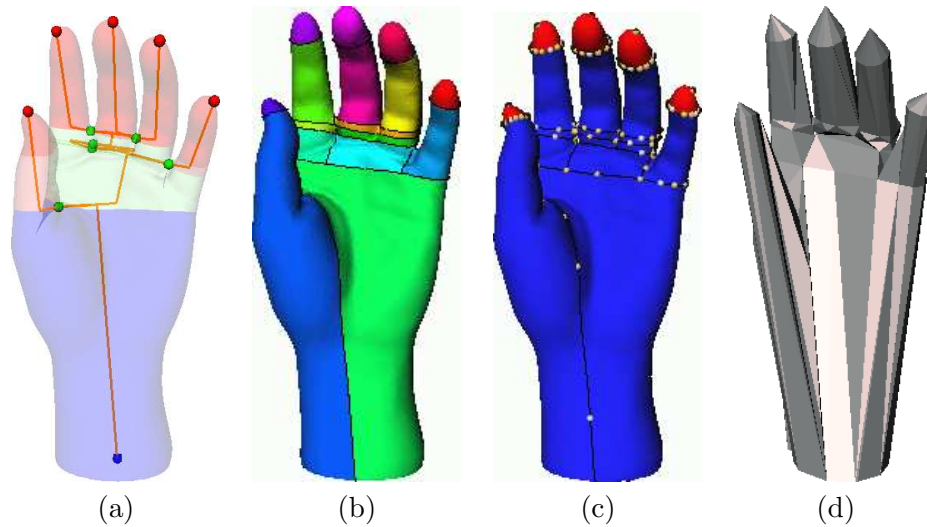
For more than ten years, size functions have been adopted as shape descriptor for binary image retrieval (1-dimensional data) because of their complex computation in case of higher dimensions. The approaches described in the dissertation extend their computation to any dimension, showing the effectiveness of this descriptor for surfaces (2-dimensional data) and volumes (3-dimensional data).

The flexibility of the choice of the real function imply that the same description framework may apply to different application contexts. Besides the application domains (i.e., shape analysis, matching, classification and retrieval) discussed in Chapter 4, our shape descriptors have been effectively used in several research projects. For instance, the shape segmentation associated to a Shape Graph has been used, together with other segmentation approaches, for semantic shape annotation. Figure 5.1 shows an example of the Shape Annotator [ARSF07] developed within the AIMSHAPE Network of Excellence [IST].

Moreover, within the SHALOM FIRB project, we are currently investigating on shape simplification and restoration driven by our Shape Graph. Figure 5.2 shows some preliminary results obtained with our tools. Once the Shape Graph has been extracted (Figure 5.2(a)), it is used to cut the shape across the arcs using isocontours of the function  $f$ . Moreover, additional lines are inserted in the mesh following the flow direction of the gradient of  $f$  in order to guarantee that all regions have only one or two boundary components, see Figure 5.2(b). Then, every boundary component of a region is sampled in order to approximate the contour (Figure 5.2(c)). Finally, the graph connectivity drives the triangle reconstruction on the basis of the points previously sampled. The mesh obtained with this process roughly approximate the original one; however, this process may be repeated until the mesh reconstructed is arbitrarily near of the original model. Therefore, this process defines a progressive mesh compression procedure.



**Figure 5.1:** The Shape Annotator is able to annotate a shape adopting different segmentations [ARSF07].



**Figure 5.2:** The Shape Graph in (a) drives the shape segmentation with iso-contours and flow lines (b). (c) The region boundaries are sampled. (d) The rough triangle mesh that approximates the original model on the basis of the point sampling.

## 5.2 Directions for future work

A number of open problems were discussed in the body of the thesis as they arose. These ranged from easy extensions of the work presented in this thesis, to potential applications, to general questions about whether we can use similar techniques to deduce other descriptors from finite data. Therefore, in the following, we will point out a number of possible developments we believe are really promising.

The techniques developed in this thesis face the definition and the development of tools able to monitorate and integrate different aspects related to shape analysis and reasoning. For instance, a concise but expressive shape description would ease the interaction with the shape resulting from the optimization process that arises in a shape design process affected by engineering constraints applied to a design space. The expert, indeed, might decide to modify the structure of the shape, according to the relevance that the various subparts have. These modifications can affect the topology of the shape. A syntetic shape description like the SG could support an interactive simplification or modification of the shape, by offering friendly tools (e.g. graph editing operations) to modify the geometry and topology of the original shape. Therefore, the required structure could be able to retain sufficient information on the original shape, keep track of the changes made, and permit a final re-check of the original conditions on the new shape.

With reference to the application of the SG to shape retrieval, we are planning to consider a multi-step approach where a set of different filters, for example coarse filters, shape harmonics and structural descriptors, are used to progressively refine the set of geometrically similar candidates. In this way we will obtain a multi-modal query mechanism that could provide a combination of various measures of shape similarity, corresponding to function, form and structure analysis of 3D shapes.

We are currently researching into the development of new real functions, in order to analyze different kind of shape features of three-dimensional models. We believe that capturing a larger amount of information would increase the retrieval performance, allowing for a better discrimination of objects, and the rejection of some of the false matchings which can be observed. Moreover, we are also investigating how the choice of the functions for the graph extraction determines the characteristics of the resulting graph configuration. The experimental results have shown that this approach is promising, and goes in the direction of developing tools to automatically annotate the shape semantic, and to encapsulate it in a digital shape representation.

We finally remark that the definition of a classification system is crucial in the field of semantic annotation. In fact, recognizing that an object belongs to a class means that its shape is expected to present a set of pre-determined features. For instance, if an object is classified as a human model, it is expected to have two arms, two legs, the body and the head. This fact suggests that a classification scheme is able to generate new knowledge

---

that can be further exploited with tools specialized in the characterization of shape sub-parts. In this sense we foresee the development of semi-automatic tools that, on the basis of knowledge techniques, like ontologies or machine learning methods, are able to simulate intelligent environments. In this context we are planning to formalize the application domain through a specific ontology which provides the rules for associating semantics to shape or shape parts [IST].

Moreover, this work can be extended and improved in several directions. The methods we have developed may apply to contexts that are more general than those presented in Chapter 4; thus, we believe that our approach to shape description could be a useful tool for analyzing data from different domains. Therefore, it would be interesting to test our method in other application fields with respect to those proposed in Chapter 4, such as bio-medicine and bio-informatics, with a particular attention to protein docking and drug design problems. In fact, owing tools able to analyse multiple aspects of the same entity promises a solution to face these problems from a multi-facet point of view, able to take into account the different levels of protein structure, the interaction between different molecules and their potential fields. In particular, this research can also impact on the visualization and simulation of human organs, where complex and huge models whose data values depend on multiple factors must be realized to predict diseases and support early diagnosis.

Other application areas can be identified in the field of physical, geo-physical and material science. Scientific data coming from physical studies typically consist of a large number of measurements taken within a domain of interest. For instance, climate scientists must couple many different simulation methods into a single "meta-simulation" that combine ocean, atmospheric, land use, vegetation, biochemistry, ecosystem dynamics, and other models. Physics simulation data sets produce hundreds of different models that require an enormous effort in terms of data management and comparative analysis. Some of these functions may be redundant, which motivates the development of comparative measures or multi-functional analysis that may be used to construct bases of functions sufficient to study the phenomena. A related application is the study of time-varying functions that model the event evolution

As further evolution of the work carried out in this dissertation, we plan to effectively contribute to the EU PROJECT *FOCUS K3D* (2008-2010) that is devoted to foster the comprehension, adoption and use of knowledge intensive technologies for coding and sharing 3D media content in consolidate and emerging application communities.





# Bibliography

- [3dc] <http://www.3dcafe.com>.
- [AAAG95] O. Aichholzer, D. Alberts, F. Aurenhammer, and B. Gartner. A novel type of skeleton for polygons. *Journal of Universal Computer Science*, 1:752–761, 1995.
- [AB04] D. Attali and J.D. Boissonnat. A linear bound on the complexity of the Delaunay triangulation of points on polyhedral surfaces. *Discrete and Computational Geometry*, 31:369–384, 2004.
- [ABL03] D. Attali, J.D. Boissonnat, and A. Lieutier. Complexity of the Delaunay triangulation of points on surfaces: the smooth case. In *SCG '03: Proc. of the 19th Annual Symposium on Computational Geometry 2003*, pages 201–210. ACM Press, 2003.
- [ABM<sup>+</sup>06] M. Attene, S. Biasotti, M. Mortara, G. Patané, M. Spagnuolo, and B. Falcidieno. Computational methods for understanding 3D shapes. *Computer & Graphics*, 30(3):323–333, 2006.
- [ABS03] M. Attene, S. Biasotti, and M. Spagnuolo. Shape understanding by contour-driven retiling. *The Visual Computer*, 19(2-3):127–138, 2003.
- [AC07] M. Allili and D. Corriveau. Topological analysis of shapes using Morse theory. *Computer Vision and Image Understanding*, 105(3):188–199, 2007.
- [Ack28] W. Ackermann. Zum hilbertschen aufbau der reellen zahlen. *Math. Ann.* 99, pages 118–133, 1928.
- [ACK01] N. Amenta, S. Choi, and R.K. Kolluri. The power crust, unions of balls, and the medial axis transform. *Computational Geometry: Theory and Applications*, 19:127–153, 2001.
- [ACZ04] M. Allili, D. Corriveau, and D. Ziou. Morse homology descriptor for shape characterization. In *ICPR2004: Proceedings of the 17<sup>th</sup> International Conference on Pattern Recognition*, volume 4, pages 27–30, 2004.
- [AdB96] C. Arcelli and G. Sanniti di Baja. Skeletons of planar patterns. In T.Y. Kong

- and A. Rosenfeld, editors, *Topological Algorithms for Digital Image Processing*, pages 99–143. North-Holland, 1996.
- [AE98] U. Axen and H. Edelsbrunner. Auditory Morse analysis of triangulated manifolds. In H. C. Hege and K. Polthier, editors, *Mathematical Visualization*, pages 223–236. Springer-Verlag, 1998.
- [AEHW04] P. K. Agarwal, H. Edelsbrunner, J. Harer, and Y. Wang. Extreme elevation on a 2-manifold. In *SCG '04: Proceedings of the 20<sup>th</sup> Annual Symposium on Computational Geometry*, pages 357–265, New York, NY, USA, 2004. ACM Press.
- [AEHW06] P. K. Agarwal, H. Edelsbrunner, J. Harer, and Y. Wang. Extreme elevation on a 2-manifold. *Discrete Comput. Geom.*, 36(4):553–572, 2006.
- [AF06] M. Attene and B. Falcidieno. ReMESH: An interactive environment to edit and repair triangle meshes. In *SMI '06: Proceedings of the IEEE International Conference on Shape Modeling and Applications 2006*, pages 271–276. IEEE Computer Society Press, 2006.
- [aim] <http://shapes.aimatshape.net/>.
- [AK00] N. Amenta and R. K. Kolluri. Accurate and efficient unions of balls. In *Proceedings of the 5<sup>th</sup> ACM Symposium on Computational Geometry*, pages 119–128. ACM Press, 2000.
- [AMT01] M. Allili, K. Mischaikow, and A. Tannenbaum. Cubical homology and the topological classification of 2D and 3D imagery. In *ICIP 2001: IEEE International Conference on Image Processing*, volume 2, pages 173–176, 2001.
- [ARSF07] M. Attene, F. Robbiano, M. Spagnuolo, and B. Falcidieno. Semantic annotation of 3D surface meshes based on feature characterization. In *Semantic and Digital Media Technologies*, LNCS, 4816, pages 126–139, 2007.
- [Aur91] F. Aurenhammer. Voronoi diagrams—A survey of a fundamental geometric data structure. *ACM Computing Surveys*, 23(3):345–405, 1991.
- [Axe99] U. Axen. Computing Morse functions on triangulated manifolds. In *SODA '99: Proceedings of the 10<sup>th</sup> Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 850–851, Philadelphia, PA, USA, 1999. ACM Press.
- [AZ03] M. Allili and D. Ziou. Computational homology approach for topology descriptors and shape representation. In *ICISP'2003: Proceedings of the International Conference on Image and Signal Processing*, volume 2, pages 508–516, 2003.
- [BA92] J. W. Brandt and V. R. Algazi. Continuous skeleton computation by Voronoi diagram. *CVGIP: Image Understanding*, 55:329–337, 1992.
- [BAB<sup>+</sup>07] S. Biasotti, D. Attali, J.-D. Boissonnat, H. Edelsbrunner, G. Elber, M. Mortara, G. Sanniti di Baja, M. Spagnuolo, and M. Tanase. Skeletal structures.
-

- In L. De Floriani and M. Spagnuolo, editors, *Shape Analysis and Structuring*, pages 145–183. 2007.
- [Ban67] T. F. Banchoff. Critical points and curvature for embedded polyhedra. *Journal of Differential Geometry*, 1:245–256, 1967.
- [Ban70] T. F. Banchoff. Critical points and curvature for embedded polyhedral surfaces. *American Mathematical Monthly*, 77(5):475–485, 1970.
- [Bau75] B. G. Baumgart. A polyhedron representation for Computer Vision. In *Proceedings of the AFIPS National Computer Conference*, volume 44, pages 589–596, 1975.
- [BBK06] A. M. Bronstein, M. M. Bronstein, and R. Kimmel. Efficient computation of isometry-invariant distances between surfaces. *SIAM Journal on Scientific Computing*, 28(5):1812–1836, 2006.
- [BCF<sup>+</sup>07] S. Biasotti, A. Cerri, P. Frosini, D. Giorgi, and C. Landi. Multidimensional size functions for shape comparison. Technical Report 04-07, IMATI, CNR, Genova (Italy), 2007.
- [BDFP07] S. Biasotti, L. De Floriani, B. Falcidieno, and L. Papaleo. Morphological representations of scalar fields. In L. De Floriani and M. Spagnuolo, editors, *Shape Analysis and Structuring*, pages 185–213. 2007.
- [BDGJ07] K. Buchin, T. K. Dey, J. Giesen, and K. John. Recursive geometry of the flow complex and topology of the flow complex filtration. *computational Geometry. Theory and Applications*, 2007.
- [BDP06] S. Berretti, A. Del Bimbo, and P. Pala. Partitioning of 3D meshes using Reeb graphs. In *ICPR '06: Proceedings of the 18<sup>th</sup> International Conference on Pattern Recognition*, pages 19–22, Washington, DC, USA, 2006. IEEE Computer Society Press.
- [BEHP03] P.-T. Bremer, H. Edelsbrunner, B. Hamann, and V. Pascucci. A multi-resolution data structure for two-dimensional Morse functions. In G. Turk, J. van Wijk, and R. Moorhead, editors, *VIS '03: Proceedings of the IEEE Visualization 2003*, pages 139–146. IEEE Computer Society Press, October 2003.
- [BEHP04] P.-T. Bremer, H. Edelsbrunner, B. Hamann, and V. Pascucci. A topological hierarchy for functions on triangulated surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 10(4):385–396, July/August 2004.
- [BFD<sup>+</sup>08] S. Biasotti, B. Falcidieno, L. De Floriani, P. Frosini, D. Giorgi, C. Landi, and M. Spagnuolo. Describing shapes by geometric-topological properties of real functions. *ACM Computing Surveys*, December 2008. in print.
- [BFS00] S. Biasotti, B. Falcidieno, and M. Spagnuolo. Extended Reeb Graphs for surface understanding and description. In G. Borgefors and G. Sanniti di Baja, editors,
-

- DGCI 2000: Proceedings of the 9<sup>th</sup> Discrete Geometry for Computer Imagery Conference*, volume 1953 of *Lecture Notes in Computer Science*, pages 185–197, Uppsala, 2000. Springer Verlag.
- [BFS02] S. Biasotti, B. Falcidieno, and M. Spagnuolo. Shape abstraction using computational topology techniques. In U. Cugini and M. Wozny, editors, *From Geometric Modeling to Shape Modeling*, pages 209–222. Kluwer Academic Publishers, 2002.
- [BFS04] S. Biasotti, B. Falcidieno, and M. Spagnuolo. Surface shape understanding based on Extended Reeb graphs. In S. Rana, editor, *Topological Data Structures for Surfaces: An Introduction for Geographical Information Science*, pages 87–103. John Wiley & Sons, 2004.
- [BG06] C. L. Bajaj and S. Goswami. Automatic fold and structural motif elucidation from 3D EM maps of macromolecules. In *ICVGIP 2006: Proceedings of the 5<sup>th</sup> Indian Conference on Computer Vision, Graphics and Image Processing*, pages 264–275, 2006.
- [BGG07] C. L. Bajaj, A. Gillette, and S. Goswami. Topology based selection and curation of level sets. In *TopoInVis 2007: Workshop on Topology-based Methods in Visualization 2007*, 2007.
- [BGM<sup>+</sup>06] S. Biasotti, D. Giorgi, S. Marini, M. Spagnuolo, and B. Falcidieno. A comparison framework for 3D classification methods. In *IW-MRCS '06*, LNCS, 4105, pages 314–321, 2006.
- [BGM<sup>+</sup>07] S. Biasotti, D. Giorgi, S. Marini, M. Spagnuolo, and B. Falcidieno. 3D classification via structural prototypes. In *Semantic and Digital Media Technologies*, LNCS, 4816, pages 140–143, 2007.
- [BGSF06] S. Biasotti, D. Giorgi, M. Spagnuolo, and B. Falcidieno. Size functions for 3D shape retrieval. In *SGP'06: Proceedings of the 2006 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, pages 239–242, 2006.
- [BGSF08a] S. Biasotti, D. Giorgi, M. Spagnuolo, and B. Falcidieno. Computing size functions for 3D models. *Pattern Recognition*, 2008. in print.
- [BGSF08b] S. Biasotti, D. Giorgi, M. Spagnuolo, and B. Falcidieno. Reeb graphs for shape analysis and applications. *Theoretical Computer Science*, 392(1–3):5–22, 2008. doi: 10.1016/j.tcs.2007.10.018.
- [BH01] G. Barequet and S. Har-Peled. Efficiently approximating the minimum-volume bounding box of a point set in three dimensions. *J. Algorithms*, 38:91–109, 2001.
- [BHH98] J. A. Bangham, J. Ruiz Hidalgo, and R. Harvey. Robust morphological scale-space trees. In Stephen Marshall, Neal Harvey, and Druti Shah, editors, *Pro-*
-

- ceedings of Noblesse Workshop on Non-Linear Model Based Image Analysis*, pages 133–139, Glasgow, UK, July 1998.
- [BHHC98] J. A. Bangham, J. R. Hidalgo, R. Harvey, and G. Cawley. The segmentation of images via scale-space trees. In *Proceedings of the 9<sup>th</sup> British Machine Vision Conference*, pages 33–43, 1998.
- [Bia04a] S. Biasotti. *Computational Topology Methods for Shape Modelling Applications*. PhD thesis, Università degli Studi di Genova, May 2004.
- [Bia04b] S. Biasotti. Reeb graph representation of surfaces with boundary. In *SMI '04: Proceedings of the International Conference on Shape Modeling and Applications 2004*, pages 371–374, Los Alamitos, June, 7-9 2004. IEEE Computer Society Press.
- [Bia05] S. Biasotti. Topological coding of surfaces with boundary using Reeb graphs. *Computer Graphics and Geometry*, 7(1):31–45, 2005.
- [Bie87] I. Biederman. Recognition-by-Components: A theory of human image understanding. *Psychological Review*, 94:115–147, 1987.
- [Bie95] I. Biederman. Visual object recognition. In S. Kosslyn and D. Osherson, editors, *An invitation to Cognitive Science*, volume 2, chapter 4, pages 121–165. MIT Press, 1995.
- [BJ85] P. J. Besl and R. C. Jain. Three-dimensional object recognition. *ACM Computing Surveys*, 17(1):75–145, 1985.
- [BJ96] E. Breen and R. Jones. Attribute openings, thinnings, and granulometries. *Computer Vision and Image Understanding*, 64:377–389, 1996.
- [BKS<sup>+</sup>05] B. Bustos, D. A. Keim, D. Saupe, T. Schreck, and D. V. Vranić. Feature-based similarity search in 3D object databases. *ACM Computing Surveys*, 37(4):345–387, December 2005.
- [BL79] S. Beucher and C. Lantuejoul. Use of watersheds in contour detection. In *Proceedings of the Int. Workshop on Image Processing: Real-Time Edge and Motion Detection/Estimation*, Rennes, France, September, 17-21 1979.
- [Blu67] H. Blum. A transformation for extracting new descriptors of shape. In W. Whaten-Dunn, editor, *Proc. of Symp. Models for Perception of Speech and Visual form*, pages 362–380. Cambridge MA: MIT Press, 1967.
- [Blu73] H. Blum. Biological shape analysis and visual science. *Journal of Theoretical Biology*, 38:205–287, 1973.
- [BM98] A. Bieniek and A. Moga. A connected component approach to the watershed segmentation. In H. Heijmans and J. Roerdink, editors, *Mathematical Morphology and its Application to Image and Signal Processing*, pages 215–222. Kluwer Acad. Publ., Dordrecht, 1998.
-

- 
- [BM00] A. Bieniek and A. Moga. An efficient watershed algorithm based on connected components. *Pattern Recognition*, 33:907–916, 2000.
- [BM05] S. Biasotti and S. Marini. 3D object comparison based on shape descriptors. *International Journal of Computer Applications in Technology*, 23(2/3/4):57–69, 2005.
- [BMM<sup>+</sup>03a] S. Biasotti, S. Marini, M. Mortara, G. Patané, M. Spagnuolo, and B. Falcidieno. 3D shape matching through topological structures. *Lecture Notes in Computer Science*, 2886:194–203, 2003.
- [BMM<sup>+</sup>03b] S. Biasotti, S. Marini, M. Mortara, G. Patané, M. Spagnuolo, and B. Falcidieno. 3D shape matching through topological structures. In I. Nyström, G. Sanniti di Baja, and S. Svensson, editors, *Proceedings of the 11<sup>th</sup> Discrete Geometry for Computer Imagery Conference*, volume 2886 of *Lecture Notes in Computer Science*, pages 194–203, Naples, 2003. Springer Verlag.
- [BMMP03] S. Biasotti, S. Marini, M. Mortara, and G. Patané. An overview on properties and efficacy of topological skeletons in shape modelling. In *Proceedings of Shape Modelling and Applications*, pages 245–254, Seoul, South Korea, May 2003. IEEE Press.
- [BMS00] S. Biasotti, M. Mortara, and M. Spagnuolo. Surface compression and reconstruction using Reeb graphs and shape analysis. In *SCCG 2000: Proceedings of the 16<sup>th</sup> Spring Conference on Computer Graphics*, pages 174–185. ACM Press, 2000.
- [BMSF06] S. Biasotti, M. Marini, M. Spagnuolo, and B. Falcidieno. Sub-part correspondence by structural descriptors of 3D shapes. *Computer Aided Design*, 38(9):1002–1019, September 2006.
- [Bor96] G. Borgefors. On digital distance transform in three dimensions. *Computer Vision and Image Understanding*, 64(3):368–376, 1996.
- [BP07] P. T. Bremer and V. Pascucci. A practical approach to two-dimensional scalar topology. In H. Hauser, H. Hagen, and H. Theisel, editors, *Topology-based Methods in Visualization*, pages 151–169. Springer Berlin Heidelberg, 2007.
- [BPH05] P.-T. Bremer, V. Pascucci, and B. Hamann. Maximizing adaptivity in hierarchical topological models. In *SMI '05: Proceedings of the International Conference on Shape Modeling and Applications 2005*, pages 300–309. IEEE Computer Society Press, 2005.
- [BPS97] C. L. Bajaj, V. Pascucci, and D. R. Schikore. The contour spectrum. In *VIS '97: Proceedings of the IEEE Visualization 1997*, pages 167–173. IEEE Computer Society Press, 1997.
- [BPS98] C. L. Bajaj, V. Pascucci, and D. R. Schikore. Visualization of scalar topology
-

- for structural enhancement. In *VIS '98: Proceedings of the IEEE Visualization 1998*, pages 51–58. IEEE Computer Society Press, 1998.
- [BPSF07] S. Biasotti, G. Patané, M. Spagnuolo, and B. Falcidieno. Analysis and comparison of real functions on triangulated surfaces. In A. Cohen, J.-L. Merrien, and L. L. Schumaker, editors, *Curve and Surface Fitting*, pages 41–50. Nashboro Press, 2007.
- [BR63] R. L. Boyell and H. Ruston. Hybrid techniques for real-time radar simulation. In *Proceedings of the 1963 Fall Joint Computer Conference*, Nov 1963.
- [BRS03] D. Bespalov, W.C. Regli, and A. Shokoufandeh. Reeb graph based shape retrieval for CAD. In *DETC'03: Proceedings of the ASME Design Engineering Technical Conferences 2003*. ASME, Sep 2003.
- [BS98] C. L. Bajaj and D. R. Schikore. Topology preserving data simplification with error bounds. *Computers and Graphics*, 22(1):3–12, 1998.
- [BS04] N. Berglund and A. Szymczak. Making contour trees subdomain-aware. In *Abstracts of the 16<sup>th</sup> Canadian Conference on Computational Geometry*, pages 188–191, 2004.
- [BTB99] V. Blanz, M. J. Tarr, and H. H. Bülthoff. What object attributes determine canonical views? *Perception*, 28:575–599, 1999.
- [Bur96] C. Burnikel. *Exact Computation of Voronoi Diagrams and Line Segment Intersections*. Ph.D thesis, Universität des Saarlandes, March 1996.
- [cae] <http://www.hec.afrl.af.mil/HECP/Card1b.shtml> #caesarsamples.
- [Car04] H. Carr. *Topological Manipulation of isosurfaces*. PhD thesis, The University of British Columbia, April 2004.
- [Cay59] A. Cayley. On Contour and Slope Lines. *London, Edinburgh and Dublin Philosophical Magazine and Journal of Science*, XVIII:264–268, 1859.
- [CB97] M. Couprie and G. Bertrand. Topological grayscale watershed transformation. In *Vision Geometry V, Proceedings of SPIE*, volume 3168, pages 136–146, 1997.
- [CBG07] A. Cerri, S. Biasotti, and D. Giorgi.  $k$ -dimensional size functions for shape description and comparison. In *ICIAP 2007: Proceedings of the 14<sup>th</sup> International Conference on Image Analysis and Processing, Modena*, September 10-14 2007. IEEE Computer Society Press.
- [CCL03] F. Cazals, F. Chazal, and T. Lewiner. Molecular shape analysis based upon the Morse-Smale complex and the Connolly function. In *SCG '03: Proceedings of the 19<sup>th</sup> Annual Symposium on Computational Geometry*, pages 351–360, New York, NY, USA, 2003. ACM Press.
- [CDF07] F. Cagliari, B. Di Fabio, and M. Ferri. One-dimensional reduction of multidimensional persistent homology. *arXiv:math/0702713*, 2007.
-

- 
- [CFG05] A. Cerri, M. Ferri, and D. Giorgi. A complete keypics experiment with size functions. In *CIVR 2005: Int. Conference on Image and Video Retrieval*, volume 3568 of *Lecture Notes in Computer Science*, pages 357–366. Springer, 2005.
  - [CFG06] A. Cerri, M. Ferri, and D. Giorgi. Retrieval of trademark images by means of size functions. *Graphical Models*, 68(5):451–471, 2006.
  - [CFP01] F. Cagliari, M. Ferri, and P. Pozzi. Size functions from the categorical viewpoint. *Acta Applicandae Mathematicae*, 67:225–235, 2001.
  - [Cga] *The CGAL 3.1 User Manual*.
  - [CKF03] J. Cox, D. B. Karron, and N. Ferdous. Topological zone organization of scalar volume data. *Journal of Mathematical Imaging and Vision*, 18:95–117, 2003.
  - [CKM04] T. Culver, J. Keyser, and D. Manocha. Exact computation of the medial axis of a polyhedron. *Computer Aided Geometric Design*, 21(1):65–98, 2004.
  - [CL03] Y.-J. Chiang and X. Lu. Progressive Simplification of Tetrahedral Meshes Preserving All Isosurface Topologies. *Computer Graphics Forum*, 22(3):493–504, 2003.
  - [CLG01] F. Cagliari, C. Landi, and L. Grasselli. Presentations of Morse homology for studying shape of manifolds. Technical Report 10, DISMI, Univ. di Modena e Reggio Emilia, 2001.
  - [CLLR05] Y.-J. Chiang, T. Lenz, X. Lu, and G. Rote. Simple and optimal output-sensitive construction of contour trees using monotone paths. *Computational Geometry: Theory and Applications*, 30:165–195, 2005.
  - [CMEH<sup>+</sup>03] K. Cole-McLaughlin, H. Edelsbrunner, J. Harer, V. Natarajan, and V. Pascucci. Loops in Reeb graphs of 2-manifolds. In *SCG '03: Proceedings of the 19<sup>th</sup> Annual Symposium on Computational Geometry*, pages 344–350, New York, NY, USA, 2003. ACM Press.
  - [COTS03] D. Chen, M. Ouhyoung, X. Tian, and Y. Shen. On visual similarity based 3D model retrieval. *Computer Graphics Forum*, 22:223–232, 2003.
  - [CS03] H. Carr and J. Snoeyink. Path seeds and flexible isosurfaces using topology for exploratory visualization. In *VISSYM '03: Proceedings of the Symposium on Data Visualisation 2003*, pages 49–58, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.
  - [CS07] H. Carr and J. Snoeyink. Representing interpolant topology for contour tree computation. In *TopoInVis 2007: Workshop on Topology-based Methods in Visualization 2007*, 2007.
  - [CSA00] H. Carr, J. Snoeyink, and U. Axen. Computing contour trees in all dimensions. In *SODA '00: Proceedings of the 11<sup>th</sup> Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 918–926, Philadelphia, PA, USA, 2000. ACM Press.
-



- 
- [CSA03] H. Carr, J. Snoeyink, and U. Axen. Computing contour trees in all dimensions. *Computational Geometry: Theory and Applications*, 24(2):75–94, 2003.
- [CSEH05] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. Stability of persistence diagrams. In *SCG '05: Proceedings of the 21<sup>st</sup> Annual Symposium on Computational Geometry*, pages 263–271, New York, NY, USA, 2005. ACM Press.
- [CSEH07a] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. Extending persistence using Poincaré and Lefschetz duality. *Foundations of Computational Mathematics*, 2007.
- [CSEH07b] D. Cohen-Steiner, H. Edelsbrunner, and J. Harer. Stability of persistence diagrams. *Discrete and Computational Geometry*, 37(1):103–120, 2007.
- [CSEM06] D. Cohen-Steiner, H. Edelsbrunner, and D. Morozov. Vines and vineyards by updating persistence in linear time. In *SCG '06: Proceedings of the 22<sup>nd</sup> Annual Symposium on Computational Geometry*, pages 119–126, New York, NY, USA, 2006. ACM Press.
- [CSM05] N. D. Cornea, D. Silver, and P. Min. Curve-Skeleton Applications. *VIS '05: Proceedings of the IEEE Visualization 2005*, pages 95–102, 2005.
- [CSM07] N. D. Cornea, D. Silver, and P. Min. Curve-skeleton properties, applications, and algorithms. *IEEE Transactions on Visualization and Computer Graphics*, 13(3):530–548, 2007.
- [CSvdP04] H. Carr, J. Snoeyink, and M. van de Panne. Simplifying flexible isosurfaces using local geometric measures. In *VIS '04: Proceedings of the IEEE Visualization 2004*, pages 497–504, 2004.
- [CSY97] T.M. Chan, J. Snoeyink, and C.K. Yap. Primal dividing and dual pruning: Output-sensitive construction of 4-d polytopes and 3-D Voronoi diagrams. *Discrete and Computational Geometry*, 18:433–454, 1997.
- [CSYB05] N. D. Cornea, D. Silver, D. Yuan, and R. Balasubramanian. Computing hierarchical curve- skeletons of 3D objects. *The Visual Computer*, 21(11):945–955, 2005.
- [Cul00] T. Culver. *Computing the medial axis of a polyhedron reliably and efficiently*. PhD thesis, University North Carolina, Chapel Hill, North Carolina, 2000.
- [CV02] S.W. Cheng and A. Vigneron. Motorcycle graphs and straight skeletons. In *SODA '02: Proc. of the 13th ACM-SIAM Symposium on Discrete Algorithms 2002*, pages 156–165. ACM Press, 2002.
- [CZCG04a] G. Carlsson, A. Zomorodian, A. Collins, and L. Guibas. Persistence barcodes for shapes. In *SGP '04: Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, pages 127–138. ACM Press, 2004.
-

- 
- [CZCG04b] A. Collins, A. Zomorodian, G. Carlsson, and L. Guibas. A barcode shape descriptor for curve point cloud data. *Computers and Graphics*, 28(6):881–894, 2004.
  - [CZCG04c] A. Collins, A. Zomorodian, G. Carlsson, and L. Guibas. A barcode shape descriptor for curve point cloud data. In *SPBG’04: Symposium on Point - Based Graphics 2004*, pages 181–191. Eurographics Association, 2004.
  - [CZCG05] G. Carlsson, A. Zomorodian, A. Collins, and L. Guibas. Persistence barcodes for shapes. *International Journal of Shape Modeling*, 11(2):149–187, 2005.
  - [d’A00] M. d’Amico. A new optimal algorithm for computing size function of shapes. In *CVPRIP Algorithms III*, Proceedings International Conference on Computer Vision, Pattern Recognition and Image Processing, pages 107–110, 2000.
  - [dBETT99] G. di Battista, P. Eades, R. Tamassia, and I. G. Tollis. *Graph drawing: Algorithms for the visualization of graphs*. Prentice-Hall, 1999.
  - [DBG<sup>+</sup>06] S. Dong, P.-T. Bremer, M. Garland, V. Pascucci, and J. Hart. Spectral surface quadrangulation. *ACM Transactions on Graphics*, 25(3), August 2006. Proceedings SIGGRAPH 2006.
  - [dBN05] G. Sanniti di Baja and I. Nyström. Skeletonization in 3D discrete binary images. In C.H. Chen and P.S.P. Wang, editors, *Handbook of Pattern Recognition and Computer Vision*, chapter 2.2, pages 137–156. World Scientific, Singapore, 3rd edition, January 2005.
  - [dBvK97] M. de Berg and M. van Kreveld. Trekking in the Alps without freezing or getting tired. *Algorithmica*, 19:306–323, 1997.
  - [DDM<sup>+</sup>03a] E. Danovaro, L. De Floriani, P. Magillo, M. M. Mesmoudi, and E. Puppo. Morphology-driven simplification and multi-resolution modeling of terrains. In E. Hoel and P. Rigaux, editors, *ACM-GIS 2003: Proceedings of the 11<sup>th</sup> International Symposium on Advances in Geographic Information Systems*, pages 63–70. ACM Press, November 2003.
  - [DDM03b] E. Danovaro, L. De Floriani, and M. M. Mesmoudi. Topological analysis and characterization of discrete scalar fields. In T. Asano, R. Klette, and C. Ronse, editors, *Theoretical Foundations of Computer Vision, Geometry, Morphology, and Computational Imaging*, volume 2616 of *Lecture Notes in Computer Science*, pages 386–402. Springer Verlag, 2003.
  - [DDV07] E. Danovaro, L. De Floriani, and M. Vitali. Multi-resolution Morse-Smale complexes for terrain modeling. In *ICIAP 2007: Proceedings of the 14<sup>th</sup> International Conference on Image Analysis and Processing*, Modena, September 10-14 ‘2007. IEEE Computer Society Press.
  - [DE95] C. J. A. Delfinado and H. Edelsbrunner. An incremental algorithm for Betti
-

- numbers of simplicial complexes on the 3-sphere. *Computer Aided Geometric Design*, 12:771–784, 1995.
- [DEG99] T. K. Dey, H. Edelsbrunner, and S. Guha. Computational Topology. In B. Chazelle, J. E. Goodman, and R. Pollack, editors, *Advances in Discrete and Computational Geometry*, volume 223 of *Contemporary Mathematics*, pages 109–143. AMS, Providence, 1999.
- [DF04a] P. Donatini and P. Frosini. Lower bounds for natural pseudodistances via size functions. *Archives of Inequalities and Applications*, 2:1–12, 2004.
- [DF04b] P. Donatini and P. Frosini. Natural pseudodistances between closed manifolds. *Forum Mathematicum*, 16(5):695–715, 2004.
- [DF07] P. Donatini and P. Frosini. Natural pseudodistances between closed surfaces. *Journal of the European Mathematical Society*, 9(2):231–253, 2007.
- [DFL99] P. Donatini, P. Frosini, and C. Landi. Deformation energy for size functions. In E. R. Hancock and M. Pelillo, editors, *EMMCVPR’99: Proceedings of the 2<sup>nd</sup> International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*, volume 1654 of *Lecture Notes in Computer Science*, pages 44–53, 1999.
- [dFL03] M. d’Amico, P. Frosini, and C. Landi. Optimal matching between reduced size functions. Technical Report 35, DISMI, Univ. di Modena e Reggio Emilia, 2003.
- [dFL05] M. d’Amico, P. Frosini, and C. Landi. Natural pseudo-distance and optimal matching between reduced size functions. Technical Report 66, DISMI, Univ. di Modena e Reggio Emilia, 2005.
- [dFL06] M. d’Amico, P. Frosini, and C. Landi. Using matching distance in size theory: A survey. *International Journal of Imaging Systems and Technology*, 16(5):154–161, 2006.
- [DFP04] F. Dibos, P. Frosini, and D. Pasquignon. The use of size functions for comparison of shapes through differential invariants. *Journal of Mathematical Imaging and Vision*, 21:107–118, 2004.
- [DFPV06] E. Danovaro, L. De Floriani, L. Papaleo, and M. Vitali. A multi-resolution representation of terrain morphology. In M. Raubal, H. Miller, A. Frank, and M. Goodchild, editors, *Geographic, Information Science*, volume 4197 of *Lecture Notes in Computer Science*. Springer, Berlin, 2006.
- [dFS04] M. d’Amico, M. Ferri, and I. Stanganelli. Qualitative asymmetry measure for melanoma detection. In *ISBI2004: Proceedings of the IEEE International Symposium on Biomedical Images*, pages 1155–1158, Arlington VA, 2004.
- [DGG03] T. K. Dey, J. Giesen, and S. Goswami. Shape segmentation and matching
-

- 
- with flow discretization. In *Algorithms and Data Structures*, volume 2748 of *Lecture Notes in Computer Science*, pages 25–36. Springer, 2003.
- [DLS07] T. K. Dey, K. Li, and J. Sun. On computing handle and tunnel loops. In *CW '07: Proc. of the IEEE International Conference on Cyberworlds, 2007*, pages 357–366, October 2007.
- [DM98] I. L. Dryden and K. V. Mardia. *Statistical shape analysis*. John Wiley & Sons New York, 1998.
- [DMP99] L. De Floriani, P. Magillo, and E. Puppo. Multi-resolution representation of shapes based on cell complexes. In G. Bertrand, M. Couprie, and L. Perroton, editors, *Discrete Geometry for Computer Imagery*, volume 1568 of *Lecture Notes in Computer Science*, pages 3–18, New York, 1999. Springer Verlag.
- [DP06] A. Del Bimbo and P. Pala. Content-based retrieval of 3D models. *ACM Transactions on Multimedia Computing, Communications and Applications*, 2(1):20–43, 2006.
- [DPS00] P. Dimitrov, C. Phillips, and K. Siddiqi. Robust and Efficient Skeletal Graphs. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1417–1423, Hilton Head, South Carolina, June 2000.
- [dR03] C. di Ruberto. Attributed skeletal graphs for shape modelling and matching. In *Proceedings of the 12<sup>th</sup> Int. Conference on Image Analysis and Processing*, pages 554–559. IEEE Press, 2003.
- [dre] <http://www.designrepository.org>.
- [DS06] T. K. Dey and J. Sun. Defining and computing curve-skeletons with medial geodesic function. In *SGP'06: Proceedings of the 2006 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*, pages 143–152, 2006.
- [dSC04] V. de Silva and G. Carlsson. Topological estimation using witness complexes. In *PBG 2004: Proceedings of the Symposium on Point-Based Graphics 2004*, pages 157–166, Zurich, 2004. Eurographics Association.
- [dSG07] V. de Silva and R. Ghrist. Coverage in sensor networks via persistent homology. *Algebraic & Geometric Topology*, 7:339–358, 2007.
- [DW07] T. K. Dey and R. Wenger. Stability of critical points with interval persistence. *Discrete and Computational Geometry*, 2007. To appear.
- [DZ03] T. K. Dey and W. Zhao. Approximating the medial axis from the voronoi diagram with a convergence guarantee. *Algorithmica*, 38(1):179–200, 2003.
- [EBG98] J. Eakins, J. Boardman, and M. Graham. Similarity retrieval of trademark images. *Multimedia*, 5(2):53–63, 1998.
- [EE99] D. Eppstein and J. Erickson. Raising roofs, crashing cycles, and playing pool: Applications of a data structure for finding pairwise interactions. *Discrete and Computational Geometry*, 22:569–592, 1999.
-

- 
- [EFL98] H. Edelsbrunner, M. Facello, and J. Liang. On the definition and the construction of pockets in macromolecules. *Discrete Applied Mathematics*, 88(1-3):83–102, 1998.
- [EH07] H. Edelsbrunner and J. Harer. Persistent homology – A survey. *Contemporary Mathematics*, 2007. To appear.
- [EHMP04] H. Edelsbrunner, J. Harer, A. Mascarenhas, and V. Pascucci. Time-varying Reeb graphs for continuous space-time data. In *SCG '04: Proceedings of the 20<sup>th</sup> Annual Symposium on Computational Geometry*, pages 366–372, New York, NY, USA, 2004. ACM Press.
- [EHNP03] H. Edelsbrunner, J. Harer, V. Natarajan, and V. Pascucci. Morse-Smale complexes for piecewise linear 3-manifolds. In *SCG '03: Proceedings of the 19<sup>th</sup> Annual Symposium on Computational Geometry*, pages 361–370. ACM Press, 2003.
- [EHZ01] H. Edelsbrunner, J. Harer, and A. Zomorodian. Hierarchical Morse complexes for piecewise linear 2-manifolds. In *SCG '01: Proceedings of the 17<sup>th</sup> Annual Symposium on Computational Geometry*, pages 70–79, New York, NY, USA, 2001. ACM Press.
- [EHZ03] H. Edelsbrunner, J. Harer, and A. Zomorodian. Hierarchical Morse-Smale complexes for piecewise linear 2-manifolds. *Discrete and Computational Geometry*, 30:87–107, 2003.
- [EK03] A. Elad and R. Kimmel. On bending invariant signatures for surfaces. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 25(10):1285–1295, 2003.
- [EL03] U. Eckhardt and L. Latecki. Topologies for the digital spaces  $\mathbb{Z}^2$  and  $\mathbb{Z}^3$ . *Computer Vision and Image Understanding*, 90:295–312, 2003.
- [ELZ00] H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topological persistence and simplification. In *IEEE Foundations of Computer Science, Proceedings 41<sup>st</sup> Annual Symposium*, pages 454–463, 2000.
- [ELZ02] H. Edelsbrunner, D. Letscher, and A. Zomorodian. Topological persistence and simplification. *Discrete Computational Geometry*, 28:511–533, 2002.
- [EM90] H. Edelsbrunner and E. P. Mücke. Simulation of Simplicity: A technique to cope with degenerate cases in geometric algorithms. *ACM Transactions on Graphics*, 9(1):66–104, 1990.
- [EMP06] H. Edelsbrunner, D. Morozov, and V. Pascucci. Persistence-sensitive simplification of functions on 2-manifolds. In *SCG '06: Proceedings of the 22<sup>nd</sup> Annual Symposium on Computational Geometry*, pages 127–134, New York, NY, USA, 2006. ACM Press.
- [ER44] C. Ehresmann and G. Reeb. Sur les champs d’éléments de contact de dimension  $p$  complètement intégrable dans une variété continuellement différentiable
-

- $v_n$ . *Comptes Rendu Hebdomadaires des Séances de l'Académie des Sciences*, 218:955–957, 1944.
- [ETA00] M. Elad, A. Tal, and S. Ar. Directed search in a 3D objects database using SVM. Technical Report HPL-2000-20R1, HP Labs, 2000.
- [ETA01] M. Elad, A. Tal, and S. Ar. Content based retrieval of VRML objects: an iterative and interactive approach. In *Proceedings of the sixth Eurographics workshop on Multimedia*, pages 107–118, 2001.
- [FAT99] I. Fujishiro, T. Azuma, and Y. Takeshima. Automating transfer function design for comprehensible volume rendering based on 3D field topology analysis. In *VIS '99: Proceedings of the IEEE Visualization 1999*, pages 467–470, 1999.
- [FF05] M. Ferri and P. Frosini. A proposal for image indexing: keypics, plastic graphical metadata. In *Internet Imaging VI, Proceedings of SPIE*, volume 5670, pages 225–231, San Jose, California, USA, 2005.
- [FK97] A. Fomenko and T. L. Kunii. *Topological Modelling for Visualization*. Springer Verlag, 1997.
- [FKS<sup>+</sup>04] T. Funkhouser, M. Kazhdan, P. Shilane, P. Min, W. Kiefer, A. Tal, S. Rusinkiewicz, and D. Dobkin. Modeling by example. *ACM Transactions on Graphics*, 23(3):652–663, 2004.
- [FKU77] H. Fuchs, Z.M. Kedem, and S.P. Uselton. Optimal surface reconstruction from planar contours. *Communications of the ACM*, 20:693–702, 1977.
- [FL97] P. Frosini and C. Landi. New pseudo-distances for the size function space. In R. A. Melter, A. Y. Wu, and L. J. Latecki, editors, *Vision Geometry VI, Proceedings of SPIE*, volume 3168, pages 52–60, 1997.
- [FL99] P. Frosini and C. Landi. Size theory as a topological tool for Computer Vision. *Pattern Recognition and Image Analysis*, 9:596–603, 1999.
- [FL01] P. Frosini and C. Landi. Size functions and formal series. *Applicable Algebra in Engineering, Communication and Computing*, 12:327–349, 2001.
- [Flo97] M. S. Floater. Parametrization and smooth approximation of surface triangulations. *Computer Aided Geometric Design*, 14(3):231–250, 1997.
- [FLP94] M. Ferri, S. Lombardini, and C. Pallotti. Leukocyte classification by size functions. In *Proceedings of the 2<sup>nd</sup> IEEE Workshop on Applications of Computer Vision*, IEEE Computer Society Press, pages 223–229, 1994.
- [FM67] H. Freeman and S. P. Morse. On searching a contour map for a given terrain profile. *Journal of the Franklin Institute*, 248:1–25, 1967.
- [FM99] P. Frosini and M. Mulazzani. Size homotopy groups for computation of natural size distances. *Bulletin of the Belgian Mathematical Society*, 6:455–464, 1999.
-

- 
- [For87] S. Fortune. A sweepline algorithm for voronoi diagrams. *Algorithmica*, 2:153–174, 1987.
- [For98] R. Forman. Morse theory for cell complexes. *Advances in Mathematics*, 134:90–145, 1998.
- [For02a] R. Forman. How many equilibria are there? An introduction to Morse theory. In *Calculus of Variations: The Rice Undergraduate Colloquium*. American Mathematical Society, 2002.
- [For02b] R. Forman. A user’s guide to Discrete Morse Theory. *Seminaire Lotharingien de Combinatoire*, 48, 2002.
- [FP99] P. Frosini and M. Pittore. New methods for reducing size graphs. *International Journal of Computer Mathematics*, 70:505–517, 1999.
- [FR99a] R. Farouki and R. Ramamurthy. Voronoi diagram and medial axis algorithm for planar domains with curved boundaries I. Theoretical foundations. *J. of Computational and Applied Mathematics*, 102:119–141, 1999.
- [FR99b] R. Farouki and R. Ramamurthy. Voronoi diagram and medial axis algorithm for planar domains with curved boundaries II. Detailed algorithm description. *J. of Computational and Applied Mathematics*, 102:119–141, 1999.
- [Fra93] J. N. Franklin. *Matrix theory*. Dover Publications, 1993.
- [Fro90] P. Frosini. A distance for similarity classes of submanifolds of a Euclidean space. *Bulletin of the Australian Mathematical Society*, 42:407–416, 1990.
- [Fro91] P. Frosini. Measuring shapes by size functions. In David P. Casasent, editor, *Intelligent Robots and Computer Vision X: Algorithms and Techniques, Proceedings of SPIE*, volume 1607, pages 122–133, Boston, MA, 1991.
- [Fro92] P. Frosini. Discrete computation of size functions. *J. Combin. Inform. System Sci.*, 17:232–250, 1992.
- [Fro96] P. Frosini. Connections between size functions and critical points. *Mathematical Methods in the Applied Sciences*, 19:555–596, 1996.
- [FS97] B. Falcidieno and M. Spagnuolo. Shape abstraction tools for modeling complex objects. In *Proceedings of Shape Modelling and Applications 1997*, pages 16–25, 1997.
- [FS98] B. Falcidieno and M. Spagnuolo. A shape abstraction paradigm for modeling geometry and semantics. In F.-E. Wolter and N. M. Patrikalakis, editors, *Proceedings of the Conference on Computer Graphics International 1998 (CGI-98)*, pages 646–657, Los Alamitos, California, June 22–26 1998. IEEE Computer Society.
- [FTAT00] I. Fujishiro, Y. Takeshima, T. Azuma, and S. Takahashi. Volume data mining using 3D field topology analysis. *IEEE Computer Graphics and Applications*, pages 46–51, 2000.
-

- 
- [GBP07] D. Giorgi, S. Biasotti, and L. Paraboschi. Watertight models track. Technical Report 09, IMATI, Genova, Italy, 2007.
- [GCO06] R. Gal and D. Cohen-Or. Salient geometric features for partial shape matching and similarity. *ACM Transactions on Graphics*, 25(1):130–150, 2006.
- [GH81] M. Greenberg and J. R. Harper. *Algebraic topology: A first course*. Addison-Wesley, 1981.
- [GHF90] C. Giertsen, A. Halvorsen, and P.R. Flood. Graph-directed modelling from serial sections. *The Visual Computer*, 6:284–290, 1990.
- [GHK99] L. Guibas, R. Holleman, and L. E. Kavraki. A probabilistic roadmap planner for flexible objects with a workspace medial axis based approach. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots Systems*, pages 254–260. IEEE Press, 1999.
- [GJ03] J. Giesen and M. John. The flow complex: a data structure for geometric modeling. In *SODA '03: Proceedings of the 14<sup>th</sup> Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 285–294, Philadelphia, PA, USA, 2003. Society for Industrial and Applied Mathematics.
- [GK00] P. Giblin and B. B. Kimia. A formal classification of 3D Medial Axis points and their local geometry. In *Proceedings of IEEE Conference on Computer Vision Pattern Recognition*, volume 1, pages 566–573, 2000.
- [GK03] P. J. Giblin and B. B. Kimia. On the local form and transitions of Symmetry Sets, Medial Axes, and Shocks. *International Journal of Computer Vision*, 54(1-3):143–157, 2003.
- [GK04] P. Giblin and B. B. Kimia. A formal classification of 3D medial axis points and their local geometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):238–251, 2004.
- [GNP<sup>+</sup>05] A. Gyulassy, V. Natarajan, V. Pascucci, P.-T. Bremer, and B. Hamann. Topology-based simplification for feature extraction from 3D scalar fields. In *VIS '05: Proceedings of the IEEE Visualization 2005*, pages 275–280. IEEE Computer Society Press, 2005.
- [GNP<sup>+</sup>06] A. Gyulassy, V. Natarajan, V. Pascucci, P.-T. Bremer, and B. Haman. A topological approach to simplification of three-dimensional scalar functions. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):474–484, 2006.
- [Gra71] A. Gramain. *Topologie des surfaces*. Presses Universitaires de France, 1971.
- [Gra03] M. Grandis. Ordinary and directed combinatorial homotopy, applied to image analysis and concurrency. *Homology Homotopy Appl.*, 5:211–231, 2003.
- [Gri76] H. B. Griffiths. *Surfaces*. Cambridge University Press, 1976.
-



- 
- [GS85] L. Guibas and J. Stolfi. Primitives for the manipulation of general subdivisions and computation of Voronoi diagrams. *ACM Transactions on Graphics*, 4(2):74–123, April 1985.
- [GSCO07] R. Gal, A. Shamir, and D. Cohen-Or. Pose-oblivious shape signature. *IEEE Transactions on Visualization and Computer Graphics*, 13(2):261–271, 2007.
- [GV89] G. Golub and G.F. VanLoan. *Matrix Computations*. John Hopkins University Press, 2nd. edition, 1989.
- [HA03] F. Hetroy and D. Attali. Topological quadrangulations of closed triangulated surfaces using the Reeb graph. *Graphical Models*, 65(1-3):131–148, 2003.
- [Har99] J. C. Hart. Using the CW-complex to represent the topological structure of implicit surfaces and solids. In *Proceedings Implicit Surfaces 1999, Eurographics/SIGGRAPH*, pages 107–112, 1999.
- [Hel01] M. Held. VRONI: An engineering approach to the reliable and efficient computation of Voronoi diagrams of points and line segments. *Computational Geometry: Theory and Applications*, 18:95–123, 2001.
- [HFS03] X. Huang, M. Fisher, and D. Smith. An efficient implementation of Max Tree with linked list and Hash table. In C. Sun, H. Talbot, S. Ourselin, and T. Adriansen, editors, *Proceedings of the VII Digital Image Computing: Techniques and Applications*, pages 299–308, Sidney, 2003.
- [HH89] J. Helman and L. Hesselink. Representation and Display of Vector Field Topology in Fluid Flow Data Sets. *Computer*, pages 27–36, 1989.
- [Hir97] M. W. Hirsch. *Differential Topology*. Springer, 1997.
- [Hir03] Anil N. Hirani. *Discrete Exterior Calculus*. PhD thesis, California Institute of Technology, May 2003.
- [HK03] B. Hamza and H. Krim. Geodesic object representation and recognition. In I. Nyström, G. Sanniti di Baja, and S. Sennson, editors, *Proceedings of the 11<sup>th</sup> Discrete Geometry for Computer Imagery Conference*, volume 2886 of *Lecture Notes in Computer Science*, pages 378–387, Naples, 2003. Springer Verlag.
- [HSKK01] M. Hilaga, Y. Shinagawa, T. Kohmura, and T. L. Kunii. Topology matching for fully automatic similarity estimation of 3D shapes. In *SIGGRAPH '01: Proceedings of the 28<sup>th</sup> Annual Conference on Computer Graphics and Interactive Techniques*, pages 203–212, Los Angeles, CA, August 2001. ACM Press.
- [HZW99] M. Handouyaya, D. Ziou, and S. Wang. Sign language recognition using moment-based size functions. In *Vision Interface 99, Trois-Rivières*, pages 210–216, 1999.
-

- 
- [IJL<sup>+</sup>05a] N. Iyer, S. Jayanti, K. Lou, Y. Kalyanaraman, and K. Ramani. Shape-based searching for product lifecycle applications. *Computer-Aided Design*, 37(13):1435–1446, 2005.
  - [IJL<sup>+</sup>05b] N. Iyer, S. Jayanti, K. Lou, Y. Kalyanaraman, and K. Ramani. Three-dimensional shape searching: state-of-the-art review and future trends. *Computer-Aided Design*, 37(5):509–530, 2005.
  - [IK94] T. Itoh and K. Koyamada. Isosurface Extraction By Using Extrema Graphs. In *VIS '94: Proceedings of Visualization '94*, pages 77–83. IEEE, 1994.
  - [IK95] T. Itoh and K. Koyamada. Automatic isosurface propagation using extrema graph and sorted boundary cell lists. *IEEE Transactions on Visualization and Computer Graphics*, 1(4):319–327, 1995.
  - [IST] EC IST. FP6 Network of Excellence: AIM@SHAPE. <http://www.aimatshape.net>. Starting date january 2004 - Duration 48 months.
  - [IYK01] T. Itoh, Y. Yamaguchi, and K. Koyamada. Fast Isosurface Generation Using the Volume Thinning Algorithm. *IEEE Transactions on Visualization and Computer Graphics*, 7(1):32–46, 2001.
  - [JA98] J. Jia and K. Abe. Automatic generation of prototypes in 3D structural object recognition. In *ICPR '98*, volume 1, pages 697–700, 1998.
  - [JBS06] M. W. Jones, J. A. Baerentzen, and M. Sramek. 3D distance fields: A survey of techniques and applications. *IEEE Transactions on Visualization and Computer Graphics*, 12(4):581–599, 2006.
  - [JK02] K. Järvelin and J. Kekäläinen. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems*, 20(4):422–446, 2002.
  - [Jol86] I. Jolliffe. *Principal Component Analysis*. Springer-Verlag, 1986.
  - [Jon99] R. Jones. Connected filtering and segmentation using component trees. *Computer Vision and Image Understanding*, 75:215–228, 1999.
  - [KFR03] M. Kazdan, T. Funkhouser, and S. Rusinkiewicz. Rotation invariant spherical harmonic representation of 3D shape descriptors. In L. Kobbelt, P. Schröder, and H. Hoppe, editors, *Proceedings of Symposium in Geometric Processing*, pages 156–165, Aachen, Germany, June 2003.
  - [KMM04] T. Kaczynski, K. Mischaikow, and M. Mrozek. *Computational Homology*, volume 157 of *Applied Mathematical Sciences*. Springer-Verlag, 2004.
  - [Koe90] J. J. Koenderink. *Solid shape*. MIT Press, Cambridge, MA, USA, 1990.
  - [KP47] J. L. Kelley and E. Pitcher. Exact Homomorphism sequences in Homology Theory. *The Annals of Mathematics*, 2<sup>nd</sup> Ser., 48(3):682–709, 1947.
-

- 
- [KR04] R. Klette and A. Rosenfeld. *Digital Geometry: Geometric Methods for Digital Picture Analysis*. Morgan Kaufmann Series in Computer Graphics and Geometric Modeling. Morgan Kaufmann, 2004.
- [KRS03] L. Kettner, J. Rossignac, and J. Snoeyink. The Safari Interface for Visualizing Time-Dependent Volume Data Using Iso-surfaces and Contour Spectra. *Computational Geometry: Theory and Applications*, 25(1-2):97–116, 2003.
- [KS00] P. Kanongchaiyos and Y. Shinagawa. Articulated Reeb graphs for interactive skeleton animation. In *Multimedia modeling: Modeling Multimedia Information and System*, pages 451–467. World Scientific, 2000.
- [KT03] S. Katz and A. Tal. Hierarchical mesh decomposition using fuzzy clustering and cuts. *ACM. Transactions on Graphics*, 22:954–961, July 2003.
- [KTZ95] B. Kimia, A. Tannenbaum, and S. Zucker. Shapes, shocks, and deformations, I: The components of shape and the reaction-diffusion space. *International Journal of Computer Vision*, 15:189–224, 1995.
- [LBM<sup>+</sup>06] D. Laney, P. T. Bremer, A. Mascarenhas, P. Miller, and V. Pascucci. Understanding the structure of the turbulent mixing layer in hydrodynamic instabilities. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1053–1060, 2006.
- [Lee82] D. T. Lee. Medial axis transform of a planar shape. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-4:363–369, 1982.
- [LF02] C. Landi and P. Frosini. Size functions as complete invariants for image recognition. In L. J. Latecki, D. M. Mount, and A. Y. Wu, editors, *Vision Geometry XI, Proceedings of SPIE*, volume 4794, pages 101–109, 2002.
- [Lie03] A. Lieutier. Any open bounded subset of  $\mathbb{R}^n$  has the same homotopy type as its medial axis. In *Proc. 8th ACM Sympos. Solid Modeling Appl.*, pages 65–75. ACM Press, 2003.
- [LLS92] L. Lam, S. W. Lee, and C. Y. Suen. Thinning Methodologies-A Comprehensive Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(9):869–885, 1992.
- [LLT03] T. Lewiner, H. Lopes, and G. Tavares. Towards optimality in discrete Morse theory. *Experimental Mathematics*, 12(3):271–285, 2003.
- [LLT04] T. Lewiner, H. Lopes, and G. Tavares. Applications of Forman’s discrete Morse theory to topology visualization and mesh compression. *IEEE Transactions on Visualization and Computer Graphics*, 10(5):499–508, 2004.
- [Lon98] S. Loncaric. A survey of shape analysis techniques-from automata to hardware. *Pattern Recognition*, 31(8):983–1001, 1998.
- [LV99] F. Lazarus and A. Verroust. Level set diagrams of polyhedral objects. In W.F. Bronsvoort and D.C. Anderson, editors, *SMA ’99: Proceedings of the*
-

- 5<sup>th</sup> *ACM Symposium on Solid Modeling and Applications 1999*, pages 130–140. ACM Press, 1999.
- [LVJ05] C. H. Lee, A. Varshney, and D. Jacobs. Mesh saliency. *ACM. Transactions on Graphics*, 24, August 2005.
- [M88] M. Mäntylä. *Introduction to Solid Modeling*. WH Freeman & Co. New York, NY, USA, 1988.
- [Mac03] D. A. Macrini. Indexing and Matching for View-Based 3-D Object Recognition using Shock Graphs. Master’s thesis, University of Toronto, 2003.
- [Mar82] D. Marr. *Vision - A computational investigation into the human representation and processing of visual information: An invitation to Cognitive Science*. W. H. Freeman, 1982.
- [Max70] J. C. Maxwell. On Hills and Dales. *The London, Edinburgh and Dublin Philosophical*, 40(269):421–425, 1870.
- [MBP07] S. Marini, S. Biasotti, and L. Paraboschi. Partial matching track on watertight models. Technical Report 10, IMATI, Genova, Italy, 2007.
- [mcg] <http://www.cim.mcgill.ca/shape/benchMark/>.
- [MD00] J. G. Mattes and J. Demongeot. Efficient algorithms to implement the confinement tree. In G. Borgefors and G. Sanniti di Baja, editors, *DGCI 2000: Proceedings of the 9<sup>th</sup> Discrete Geometry for Computer Imagery Conference*, volume 1953 of *Lecture Notes in Computer Science*, pages 392–405, Uppsala, 2000. Springer Verlag.
- [MD07] M. M. Mesmoudi and L. De Floriani. Morphology-based representations of discrete scalar fields. In *GRAPP 2007: International Conference on Computer Graphics Theory*, pages 137–144, Barcelona, Spain, 2007.
- [MDD<sup>+</sup>07] P. Magillo, E. Danovaro, L. De Floriani, L. Papaleo, and M. Vitali. Extracting terrain morphology: A new algorithm and a comparative evaluation. In *Proceedings of the 2<sup>nd</sup> International Conference on Computer Graphics Theory and Applications*, March 8-11 ‘2007.
- [MDDP07] M. M. Mesmoudi, E. Danovaro, L. De Floriani, and U. Port. Surface segmentation through concentrated curvature. In *ICIAP 2007: Proceedings of the 14<sup>th</sup> International Conference on Image Analysis and Processing*, Modena, September 10-14 ‘2007. IEEE Computer Society Press.
- [Mer73] R. D. Merrill. Representation of contours and regions for efficient computer search. *Communications of the ACM*, 16(2):69–82, 1973.
- [Mey94] F. Meyer. Topographic distance and watershed lines. *Signal Processing*, 38:113–125, 1994.
- [MG00] P. Monasse and F. Guichard. Fast computation of a contrast-invariant image representation. *IEEE Transactions on Image Processing*, 9(5):860–872, 2000.
-

- 
- [Mil63] J. Milnor. *Morse Theory*. Princeton University Press, New Jersey, 1963.
- [Mil65] J. Milnor. *Lectures on  $h$ -cobordism*. Princeton University Press, 1965.
- [MM05] S. Mizuta and T. Matsuda. Description of Digital Images by Region-based Contour Trees. *Lecture Notes in Computer Science*, 3656:549–558, 2005.
- [Moh91] B. Mohar. The Laplacian spectrum of graphs. *Graph Theory, Combinatorics, and Applications*, 2:871–898, 1991.
- [Mor86] M. E. Mortenson. *Geometric Modeling*. John Wiley & Sons, 1986.
- [MP02] M. Mortara and G. Patané. Shape-covering for skeleton extraction. *International Journal of Shape Modelling*, 8(2):245–252, 2002.
- [MPS<sup>+</sup>04] M. Mortara, G. Patané, M. Spagnuolo, B. Falcidieno, and J. Rossignac. Blowing bubbles for multi-scale analysis and decomposition of triangle meshes. *Algorithmica*, 38(1):227–248, 2004.
- [MR96] A. Meijster and J. Roerdink. Computation of watersheds based on parallel graph algorithms. In P. Maragos, R. W. Shafer, and M. A. Butt, editors, *Mathematical Morphology and its Application to Image Segmentation*, pages 305–312. Kluwer, 1996.
- [MSF07] S. Marini, M. Spagnuolo, and B. Falcidieno. Structural shape prototypes for the automatic classification of 3d objects. *IEEE Computer Graphics and Applications*, 27(4), 2007.
- [MSS92] D. Meyers, S. Skinner, and K. Sloan. Surfaces from contours. *ACM Transactions on Graphics*, 11:228–258, 1992.
- [Mun00] J. Munkres. *Topology*. Prentice Hall, 2000.
- [MW99] A. Mangan and R. Whitaker. Partitioning 3D surface meshes using watershed segmentation. *IEEE Transaction on Visualization and Computer Graphics*, 5(4):308–321, 1999.
- [Nac84] L.R. Nackman. Two-dimensional Critical Point Configuration Graphs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(4):442–450, 1984.
- [NB05] M. Neuhaus and H. Bunke. Graph-based multiple classifier systems a data level fusion approach. In *ICIAP 2005*, LNCS, 3617, pages 479–486, 2005.
- [NC03] L. Najman and M. Couprie. Watershed algorithms and contrast preservation. In Ingela Nyström, Gabriella Sanniti di Baja, and Stina Svensson, editors, *DGCI 2003: Proceedings of the 11<sup>th</sup> Discrete geometry for computer imagery*, volume 2886 of *Lecture Notes in Computer Science*, pages 62–71, Naples, 2003. Springer.
-

- 
- [NGH04] X. Ni, M. Garland, and J. C. Hart. Fair Morse functions for extracting the topological structure of a surface mesh. *ACM Transaction on Graphics*, 23(3):613–622, 2004.
- [NP85] L. R. Nackman and S. Pizer. Three-dimensional shape description using the Symmetric Axis Transform I: Theory. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-7:187–201, March 1985.
- [NP05] V. J. Natarajan and V. Pascucci. Volumetric data analysis using Morse-Smale complexes. In *SMI '05: Proceedings of the International Conference on Shape Modeling and Applications 2005*, pages 320–325, June 2005.
- [NS96] L. Najman and M. Schmitt. Geodesic Saliency of Watershed Contours and Hierarchical Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(12):1163–1173, 1996.
- [NSW06] P. Niyogi, S. Smale, and S. Weinberger. Finding the homology of submanifolds with high confidence from random samples. *Discrete and Computational Geometry*, 2006.
- [NWB<sup>+</sup>06] V. Natarajan, Y. Wang, P.-T. Bremer, V. Pascucci, and B. Hamann. Segmenting molecular surfaces. *Computer Aided Geometric Design*, 23(6):495–509, 2006.
- [Ogn94a] R. L. Ogniewicz. Skeleton-space: A multi-scale shape description combining region and boundary information. In *CVPR '94: Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition 1994*, pages 746–751, Los Alamitos, 1994. IEEE Computer Society.
- [Ogn94b] R. L. Ogniewicz. Skeleton-space: A multi-scale shape description combining region and boundary information. In *Proceedings of Comput. Vision Pattern Recogn.*, pages 746–751, 1994.
- [OK95] R. L. Ogniewicz and O. Kubler. Hierarchic Voronoi skeletons. *Pattern Recognition*, 28:343–359, 1995.
- [OPC96] J. M. Oliva, M. Perrin, and S. Coquillart. 3D reconstruction of complex polyhedral shapes from contours using a simplified generalised Voronoi diagram. *Computer Graphics Forum*, 15:307–318, 1996.
- [Pag03] D. L. Page. *Part Decomposition of 3D Surfaces*. PhD thesis, University of Tennessee, Knoxville, May 2003.
- [Pas04] V. Pascucci. Topology diagrams of scalar fields in scientific visualization. In S. Rana, editor, *Topological Data Structures for Surfaces*, pages 121–129. John Wiley & Sons Ltd, 2004.
- [Pav95] T. Pavlidis. A review of algorithms for shape analysis. *Document image analysis table of contents*, pages 145–160, 1995.
-

- 
- [PBF07a] L. Paraboschi, S. Biasotti, and B. Falcidieno. 3D scene comparison using topological graphs. In *Proceedings 5th Eurographics Italian Chapter Conference*, pages 87–93, Trento, 2007. The Eurographics Association.
- [PBF07b] L. Paraboschi, S. Biasotti, and B. Falcidieno. Comparing sets of 3D digital shapes through topological structures. In F. Escolano and M. Vento, editors, *Gbr2007: Proceedings of Graph-based Representations in Pattern Recognition*, volume 4538 of *Lecture Notes in Computer Science*, pages 114–125, Alicante, 2007. Springer Verlag.
- [PCM02] V. Pascucci and K. Cole-McLaughlin. Efficient computation of the topology of the level sets. In *VIS '02: Proceedings of the IEEE Visualization 2002*, pages 187–194. IEEE Press, 2002.
- [PCM03] V. Pascucci and K. Cole-McLaughlin. Parallel computation of the topology of level sets. *Algorithmica*, 38(1):249–268, 2003.
- [PCMS04] V. Pascucci, K. Cole-McLaughlin, and G. Scorzelli. Multi-resolution computation and presentation of contour trees. Technical report, LLNL Technical Report number UCRL-PROC-208680, 2004.
- [PDP06] E. Pekalska, R. P. W. Duin, and P. Paclik. Prototype selection for dissimilarity-based classifiers. *Pattern Recognition*, 39(2):189–208, 2006.
- [Pfa76] J. L. Pfaltz. Surface networks. *Geographical Analysis*, 8:77–93, 1976.
- [PGJA03] S. Pizer, G. Geric, S. Joshi, and S. R. Aylward. Multiscale medial shape-based analysis of image objects. In *Proceedings of the IEEE*, volume 91, pages 1670–1679, 2003.
- [PKS<sup>+</sup>03] D. L. Page, A. Koschan, S. R. Sukumar, B. Roui-Abidi, and M. A. Abidi. Shape analysis algorithm based on information theory. In *Proceedings of the IEEE International Conference on Image Processing*, volume I, pages 229–232, 2003.
- [PM82] J. Palis and W. De Melo. *Geometric Theory of Dynamical Systems: An Introduction*. Springer-Verlag, 1982.
- [PP93] U. Pinkall and K. Polthier. Computing discrete minimal surfaces and their conjugates. *Experimental Mathematics*, pages 15–36, 1993.
- [PPG<sup>+</sup>05] O. Polonsky, G. Patané, C. Gotsman, S. Biasotti, and M. Spagnuolo. What’s in an image? towards the computation of the ”best” view of an object. *The Visual Computer*, 21(8-10):840–847, 2005.
- [PRC81] S. E. Palmer, E. Rosch, and P. Chase. Canonical perspective and the perception of objects. *Attention and Performance*, IX:135–151, 1981.
- [pri] <http://shape.cs.princeton.edu/benchmark/>.
-

- 
- [PSBM07] V. Pascucci, G. Scorzelli, P.-T. Bremer, and A. Mascarenhas. Robust on-line computation of Reeb graphs: simplicity and speed. *ACM Transactions on Graphics*, 26(3), August 2007. SIGGRAPH 2007: Proceedings of the 34<sup>th</sup> Annual Conference on Computer Graphics and Interactive Techniques.
- [PSF04] G. Patané, M. Spagnuolo, and B. Falcidieno. Graph-based parameterization of triangle meshes with arbitrary genus. *Computer Graphics Forum*, 23(4):783–797, 2004.
- [Ran04] S. Rana, editor. *Topological Data Structures for Surfaces: An Introduction for Geographical Information Science*. John Wiley & Sons, Europe, London, 2004.
- [Ree46] G. Reeb. Sur les points singuliers d’une forme de Pfaff complètement intégrable ou d’une fonction numérique. *Comptes Rendus Hebdomadaires des Séances de l’Académie des Sciences*, 222:847–849, 1946.
- [Req80] A. Requicha. Representations of rigid solids: Theory, methods and systems. *ACM Computing Surveys*, 12(4):437–464, 1980.
- [RG02] M. Ramanathan and B. Gurumoorthy. Constructing medial axis transform of planar domains with curved boundaries. *CAD*, 35:619–632, 2002.
- [RM00] J. Roerdink and A. Meijster. The watershed transform: definitions, algorithms, and parallelization strategies. *Fundamenta Informaticae*, 41:187–228, 2000.
- [Rob99] V. Robins. Towards computing homology from finite approximations. In W.W. Comfort, R. Heckmann, R.D. Kopperman, and L. Narici, editors, 14<sup>th</sup> *Summer Conference on General Topology and its Applications*, volume 24 of *Topology Proceedings*, pages 503–532, 1999.
- [Rob02] V. Robins. Computational topology for point data: Betti numbers of  $\alpha$ -shapes. In K. Mecke and D. Stoyan, editors, *Morphology of Condensed Matter: Physics and Geometry of Spatially Complex Systems*, volume 600 of *Springer LNP*, pages 261–274, 2002.
- [Ros98] A. Rosenfield. Digital geometry: Introduction and Bibliography. In R. Klette, A. Rosenfield, and F. Sloboda, editors, *Advances in Digital and Computational Geometry*, chapter 1. Springer-Verlag, 1998.
- [RWP06] M. Reuter, F.-E. Wolter, and N. Peinecke. Laplace-Beltrami spectra as Shape-DNA of surfaces and solids. *Computer-Aided Design*, 38(4):342–366, 2006.
- [SAR96] D. Sheehy, C. Armstrong, and D. Robinson. Shape description by medial axis construction. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):62–72, 1996.
- [SB06] B.-S. Sohn and C. L. Bajaj. Time-varying contour topology. *IEEE Transactions on Visualization and Computer Graphics*, 12(1):14–25, 2006.
- [SBC<sup>+</sup>05] I. Stanganelli, A. Brucale, L. Calori, R. Gori, A. Lovato, S. Magi, B. Kopf, R. Bacchilega, V. Rapisarda, A. Testori, P. A. Ascierio, E. Simeone, and
-



- M. Ferri. Computer-aided diagnosis of melanocytic lesions. *Anticancer Research*, 25(6C):4577–4582, 2005.
- [SBTZ02] K. Siddiqi, S. Bouix, A. Tannenbaum, and S. W. Zucker. Hamilton-jacobi skeletons. *International Journal of Computer Vision*, 48(3):215–231, July 2002.
- [Sch05] B. Schneider. Extraction of hierarchical surface networks from bilinear surface patches. *Geographical Analysis*, 37:244–263, 2005.
- [SERB99] A. Sheffer, M. Etzion, A. Rappoport, and M. Bercovier. Hexahedral mesh generation using the embedded Voronoi graph. *Engineering Comput.*, 15:248–262, 1999.
- [SF01] D. Steiner and A. Fischer. Topology recognition of 3D closed freeform objects based on topological graphs. In *Pacific Graphics*, pages 82–88, 2001.
- [Sha06] A. Shamir. Segmentation and Shape Extraction of 3D Boundary Meshes. pages 137–149, Vienna, Austria, September 2006. Eurographics Association.
- [SK91] Y. Shinagawa and T.L. Kunii. Constructing a Reeb graph automatically from cross sections. *IEEE Computer Graphics and Applications*, 11:44–51, 1991.
- [SKK91] Y. Shinagawa, T. L. Kunii, and Y. L. Kergosien. Surface coding based on Morse theory. *IEEE Computer Graphics and Applications*, 11:66–78, 1991.
- [SKK04] T. B. Sebastian, P. N. Klein, and B. B. Kimia. Recognition of shapes by editing their Shock Graphs. *IEEE Transactions on Pattern Recognition and Machine Intelligence*, 26(5):550–571, 2004.
- [SL01] D. Shattuck and R. Leahy. Automated graph based analysis and correction of cortical volume topology. *IEEE Transactions on Medical Imaging*, 20:1167–1177, 2001.
- [SOG98] P. Salembier, A. Oliveras, and L. Garrido. Antiextensive connected operators for image and sequence processing. *IEEE Transactions on Image Processing*, 7:555–570, Apr 1998.
- [SP07] K. Siddiqi and S. M. Pizer, editors. *Medial Representations: Mathematics, Algorithms and Applications*. Springer, 2007.
- [Spa66] E. H. Spanier. *Algebraic Topology*. McGraw Hill, 1966.
- [Spa97] M. Spagnuolo. *Shape Abstraction tool for Modelling and Analysing natural Surfaces*. PhD thesis, Institute Nationale des Sciences Appliquées, Lyon, France, Décembre 1997.
- [SPB96] E. C. Sherbrooke, N. M. Patrikalakis, and E. Brisson. An algorithm for the medial axis transform of 3D polyhedral solids. *IEEE Transactions on Visualization and Computer Graphics*, 22:44–61, 1996.
-

- 
- [SPW96] E. Sherbrooke, N. M. Patrikalakis, and F.-E. Wolter. Differential and topological properties of medial axis transforms. *Graphical Models and Image Processing*, 58:574–592, 1996.
  - [SS00] S. L. Stoev and W. Strasser. Extracting Regions of Interest Applying a Local Watershed Transformation. In *VIS 2000: Proceedings of the IEEE Visualization 2000*, pages 21–28, Los Alamitos, 2000. IEEE Computer Society Press.
  - [SSDZ98] K. Siddiqi, A. Shokoufandeh, S.J. Dickenson, and S.W. Zucker. Shock graphs and shape matching. In *Proceedings of the 5<sup>th</sup> International Conference on Computer Vision*, pages 222–229, 1998.
  - [SSGD03] H. Sundar, D. Silver, N. Gagvani, and S. Dickinson. Skeleton based shape matching and retrieval. In *Proceedings of Shape Modelling and Applications*, pages 130–139, Seoul, South Korea, June 2003. IEEE Press.
  - [SSK<sup>+</sup>05] V. Surazhsky, T. Surazhsky, D. Kirsanov, S. J. Gortler, and H. Hoppe. Fast exact and approximate geodesics on meshes. *ACM Transactions on Graphics*, 24(3):553–560, 2005.
  - [sta] <http://graphics.stanford.edu/data/3Dscanrep/>.
  - [STG<sup>+</sup>97] D. Storti, G. Turkiyyah, M. Ganter, C. Lim, and D. Stal. Skeleton-based modeling operations on solids. In *Proceedings of the ACM Symposium on Solid Modeling and Applications*, pages 68–75. ACM Press, 1997.
  - [SV01] D. Saupe and D. V. Vranic. 3D model retrieval with spherical harmonics and moments. In *Proceedings of the 23<sup>rd</sup> DAGM-Symposium on Pattern Recognition*, pages 392–397, London, UK, 2001. Springer-Verlag.
  - [SW04] B. Schneider and J. Wood. Construction of metric surface networks from raster-based DEMs. In S. Rana, editor, *Topological Data Structures for Surfaces: An Introduction for Geographical Information Science*, pages 53–70. John Wiley & Sons Ltd, 2004.
  - [Szy05] A. Szymczak. Subdomain aware contour trees and contour evolution in time-dependent scalar fields. In *SMI '05: Proceedings of the International Conference on Shape Modeling and Applications 2005*, pages 136–144. IEEE Computer Society Press, June 2005.
  - [Tak04] S. Takahashi. *Algorithms for Extracting Surface Topology from Digital Elevation Models*, pages 31–51. John Wiley & Sons Ltd, 2004.
  - [TFT05] S. Takahashi, I. Fujishiro, and Y. Takeshima. Interval volume decomposer: A topological approach to volume traversal. In R. F. Erbacher, K. Börner, M. Gröhn, and J. C. Roberts, editors, *Visualization and Data Analysis 2005, Proceedings of SPIE*, volume 5669, pages 103–114, 2005.
  - [TIS<sup>+</sup>95] S. Takahashi, T. Ikeda, Y. Shinagawa, T. L. Kunii, and M. Ueda. Algorithms for extracting correct critical points and constructing topological graphs from
-

- discrete geographic elevation data. *Computer Graphics Forum*, 14(3):181–192, 1995.
- [TNTF04] S. Takahashi, G. M. Nielson, Y. Takeshima, and I. Fujishiro. Topological volume skeletonization using adaptive tetrahedralization. In *Geometric Modelling and Processing 2004*, pages 227–236, 2004.
- [TRW07] H. Theisel, C. Roessl, and T. Weinkau. Morphological representations of vector fields. In L. De Floriani and M. Spagnuolo, editors, *Shape Analysis and Structuring*, pages 215–240. 2007.
- [TS04] T. Tung and F. Schmitt. Augmented Reeb graphs for content-based retrieval of 3D mesh models. In *SMI '04: Proceedings of the International Conference on Shape Modeling and Applications 2004*, pages 157–166, Los Alamitos, June 2004. IEEE Computer Society Press.
- [TS05] T. Tung and F. Schmitt. The Augmented Multiresolution Reeb Graph approach for content-based retrieval of 3D shapes. *International Journal of Shape Modelling*, 11(1):91–120, June 2005.
- [tsi] <http://www.tsi.enst.fr/3dmodels/>.
- [TTF04] S. Takahashi, Y. Takeshima, and I. Fujishiro. Topological volume skeletonization and its application to transfer function design. *Graphical Models*, 66(1):24–49, 2004.
- [Tur98] M. J. Turner. Design of entropy conscious and constrained image operations using a contour tree image definition”. In *2<sup>nd</sup> IMA Conference on Image Processing: Mathematical Methods, Algorithms and Applications (Sept. 22-25, 1998)*, 1998.
- [TV98] S. P. Tarasov and M. N. Vyalyi. Construction of contour trees in 3D in  $O(n \log n)$  steps. In *SCG '98: Proceedings of the 14<sup>th</sup> Annual Symposium on Computational Geometry*, pages 68–75. ACM Press, 1998.
- [TV04a] M. Tanase and R. C. Veltkamp. A straight skeleton approximating the medial axis. *Lecture Notes in Computer Science*, 3221:809–821, Sep 2004.
- [TV04b] J. W. H. Tangelder and R. C. Veltkamp. A survey of content based 3D shape retrieval methods. In *SMI '04: Proceedings of the IEEE International Conference on Shape Modeling and Applications 2004*, pages 145–156, 2004.
- [TvL84] R. E. Tarjan and J. von Leeuwen. Worst-case analysis of set union algorithms. *Journal of the Association for Computing Machinery*, 31(2):245–281, 1984.
- [UV94] C. Uras and A. Verri. On the recognition of the alphabet of the sign language through size functions. In *Proceedings of the 12<sup>th</sup> IAPR International Conference on Pattern Recognition, Jerusalem (Israel)*, volume II, pages 334–338, Los Alamitos, CA, 1994. IEEE Computer Society Press.
-

- 
- [UV97] C. Uras and A. Verri. Computing size functions from edge maps. *International Journal of Computer Vision*, 23(2):169–183, 1997.
- [Veg97] G. Vegter. Computational Topology. In J. E. Goodman and J. O’Rourke, editors, *Handbook of Discrete and Computational Geometry*, pages 517–536. CRC Press, Boca Raton, New York, 1997.
- [VFSH01] P.-P. Vázquez, M. Feixas, M. Sbert, and W. Heidrich. Viewpoint selection using viewpoint entropy. In *Proceedings of VMV*, pages 273–280, 2001.
- [VH01] R. C. Veltkamp and M. Hagendoorn. State-of-the-Art in shape matching. In M. Lew, editor, *Principles of Visual Information Retrieval*, pages 87–119. Springer-Verlag, 2001.
- [vKvOB<sup>+</sup>97] M. van Kreveld, R. van Oostrum, C. L. Bajaj, V. Pascucci, and D. Schikore. Contour trees and small seed sets for isosurface traversal. In *SCG ’97: Proceedings of the 13<sup>th</sup> Annual Symposium on Computational Geometry*, pages 212–220, New York, NY, USA, 1997. ACM Press.
- [vKvOB<sup>+</sup>04] M. van Kreveld, R. van Oostrum, C. L. Bajaj, V. Pascucci, and D. Schikore. Contour trees and small seed sets for isosurface generation. In S. Rana, editor, *Topological Data Structures for Surfaces: An Introduction for Geographical Information Science*, pages 71–86. John Wiley & Sons, 2004.
- [VL00] A. Verroust and F. Lazarus. Extracting skeletal curves from 3D scattered data. *The Visual Computer*, 16:15–25, 2000.
- [VS91] L. Vincent and P. Soille. Watershed in digital spaces: An efficient algorithm based on immersion simulation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):583–598, 1991.
- [VT03] R. C. Veltkamp and J. W. Tangelder. Polyhedral model retrieval using weighted point sets. In *Proceedings of Shape Modelling and Applications*, pages 119–128, Seoul, South Korea, 2003. IEEE Press.
- [VU94] A. Verri and C. Uras. Invariant size functions. In J. L. Mundy, A. Zisserman, and D. Forsyth, editors, *Applications of Invariance in Computer Vision*, volume 825 of *Lecture Notes in Computer Science*, pages 215–234. Springer, 1994.
- [VU96] A. Verri and C. Uras. Metric-topological approach to shape representation and recognition. *Image and Vision Computing*, 14:189–207, 1996.
- [VUFF93] A. Verri, C. Uras, P. Frosini, and M. Ferri. On the use of size functions for shape analysis. *Biological Cybernetics*, 70:99–107, 1993.
- [VY90] G. Vegter and C. K. Yap. Computational complexity of combinatorial surfaces. In *Proceedings of the 6<sup>th</sup> Annual ACM Symposium on Computational Geometry (SOCG’90)*, pages 102–111. ACM Press, 1990.
-

- 
- [WAB<sup>+</sup>05] Y. Wang, P.K. Agarwal, P. Brown, H. Edelsbrunner, and J. Rudolph. Coarse and reliable geometric alignment for protein docking. In *Proceedings of Pacific Symposium on Biocomputing 2005*, 2005.
- [WDC<sup>+</sup>07] G. H. Weber, S. E. Dillard, H. Carr, V. Pascucci, and B. hamann. Topology-controlled volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 13(2):330–341, 2007.
- [WDSB00] Z. J. Wood, M. Desbrun, P. Schroeder, and D. Breen. Semi-regular mesh extraction from volumes. In *VIS '02: Proceedings of the IEEE Visualization 2002*, pages 275–282, Los Alamitos, 2000. IEEE Computer Society Press.
- [Wer94] J. Wernecke. *The Inventor Mentor*. Addison–Wesley, Reading, MA, 1994.
- [WHDS04] Z. Wood, H. Hoppe, M. Desbrun, and P. Schroeder. Removing excess topology from isosurfaces. *ACM Transactions on Graphics*, 23:190–208, 2004.
- [WHL05] R. C. Wilson, E. R. Hancock, and B. Luo. Pattern vectors from algebraic graph theory. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(7):1112–1124, 2005.
- [Wil70] S. Willard. *General topology*. Addison-Wesley Publishing Company, 1970.
- [WLK<sup>+</sup>02] M. Wan, Z. Liang, Q. Ke, L. Hong, I. Bitter, and A. Kaufman. Automatic centerline extraction for virtual colonoscopy. *IEEE Trans. on Medical Imaging*, 21(12):1450–1460, December 2002.
- [Wol92] F.E. Wolter. Cut locus & medial axis in global shape interrogation & representation. Technical Report Design Laboratory Memorandum 92-2, MIT, 1992.
- [Wol04] G. W. Wolf. Topographic surfaces and surface networks. In S. Rana, editor, *Topological Data Structures for Surfaces*, pages 15–29. John Wiley & Sons Ltd, 2004.
- [WS04] G. H. Weber and G. Scheuermann. Automating transfer function design based on topology analysis. In G. Brunnett, B. Hamann, H. Mueller, and L. Linsen, editors, *Geometric Modeling for Scientific Visualization*. Springer, 2004.
- [WSH03] G. H. Weber, G. Scheuermann, and B. Hamann. Detecting critical regions in scalar fields. In G.-P. Bonneau, S. Hahmann, and C. D. Hansen, editors, *VISSYM '03: Proceedings of the Symposium on Data Visualisation 2003*, pages 85–94, New York, New York, 2003.
- [WSHH02] G. H. Weber, G. Schueuermann, H. Hagen, and B. Hamann. Exploring scalar fields using critical isovalues. In *VIS '02: Proceedings of the IEEE Visualization 2002*, pages 171–178. IEEE Computer Society Press, 2002.
- [WW97] D. Weinshall and M. Werman. On view likelihood and stability. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(2):97–108, 1997.
-

- 
- [WXS06] N. Werghi, Y. Xiao, and J. P. Siebert. A functional-based segmentation of human body scans in arbitrary postures. *IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics*, 36(1):153–165, 2006.
- [XSW03] Y. Xiao, P. Siebert, and N. Werghi. A discrete Reeb graph approach for the segmentation of human body scans. In *3DIM 2003: Proceedings of the 4<sup>th</sup> International Conference on 3-D Digital Imaging and Modeling*, pages 378–385, Los Alamitos, 2003. IEEE Computer Society Press.
- [ZA02] D. Ziou and M. Allili. Generating cubical complexes from image data and computation of the Euler number. *Pattern Recognition*, 35:2833–2839, 2002.
- [ZBB04] X. Zhang, C. L. Bajaj, and N. Baker. Fast Matching of Volumetric Functions Using Multi-resolution Dual Contour Trees. Technical report, Texas Institute for Computational and Applied Mathematics, Austin, Texas, 2004.
- [ZC04] A. Zomorodian and G. Carlsson. Computing persistent homology. In *SCG '04: Proceedings of the 20<sup>th</sup> Annual Symposium on Computational Geometry*, pages 255–262, New York, NY, USA, 2004. ACM Press.
- [ZC05] A. Zomorodian and G. Carlsson. Computing persistent homology. *Discrete and Computational Geometry*, 33(2):249–274, 2005.
- [ZC07] A. Zomorodian and G. Carlsson. Localized homology. In *SMI '07: Proceedings of the IEEE International Conference on Shape Modeling and Applications 2007*, pages 189–198. IEEE Computer Society Press, 2007.
- [ZMT05] E. Zhang, K. Mischaikow, and G. Turk. Feature-based surface parameterization and texture mapping. *ACM Transaction on Graphics*, 24(1):1–27, 2005.
- [Zom05] A. Zomorodian. *Topology for Computing*. Cambridge University Press, New York, NY, 2005.
- [ZP01] T. Zaharia and F. J. Preteux. 3D-shape-based retrieval within the MPEG-7 framework. In E. R. Dougherty and J. T. Astola, editors, *Nonlinear Image Processing and Pattern Analysis XII, Edward R. Dougherty; Jaakko T. Astola; Eds.*, volume 4304 of *Proc. of the Society of Photo-Optical Instrumentation Engineers (SPIE) Conference*, pages 133–145, May 2001.
- [ZSM<sup>+</sup>05] J. Zhang, K. Siddiqi, D. Macrini, A. Shokoufandeh, and S. Dickinson. Retrieving articulated 3-D models using medial surfaces and their graph spectra. In *Proceedings of International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR, St. Augustine, FL, November 2005*.
- [ZTS02] E. Zuckerberger, A. Tal, and S. Shlafman. Polyedral surface decomposition with applications. *Computers & Graphics*, 26:733–743, 2002.
-